Quality
Development
Program

DESIGN FOR RELIABILITY
Course Outline

Day One - Monday

8:30 1.) Attributes of Reliable Computing and Their Definitions (1 Hr)
- -Stages in the Life of a System
- -Hardware reliability, generic definitions of reliability and availability
- -software reliability
- -System reliability and availability

9:30 2.) Why Design for Reliability (1/2 Hr)
- -Overview of Financial Models
- -Overview of Market Models

10:00 3.) Fundamental Reliability Theory
3.1) Faults and Their Manifestations (1 Hr)
- -Cause/Effect Sequence of Faults
- -Permanent Faults and Their Models
  - --Bath Tub Curve
- -Transient Faults and Their Models
- -Example faults and data on System Level Faults

11:00 3.2) Hardware Fault Prediction (1 1/2 Hr)
- -Mil 217 models and predictions
- -IC failure mechanisms (plus Arrhenius Model)
- -Introduction to a Mil 217 modeling program

1:30 3.3) Software Fault Prediction (1 Hr)
- -Example software bug models

2:30 Laboratory: Prediction Tools to Model the VAX 11/780 (2 Hr)

Day Two - Tuesday

8:30 Guest Speaker: John Shebell - Reliability and its Role in DEC Field Service (1 Hr)

9:30 3.4) System's Design (2 Hr)
- -Terminology
- -Stages in Fault Handling
- -The Hardware/Software Continum in Fault Handling
- -Fault Avoidance
  - --Examples of Fault Avoidance
  - --Temperature effects on reliability
  - --Part selection
- -Overview of Hardware Fault Detection Techniques
  - --Table 3.1 page 67 in [Siewiorek and Swarz]
- -Overview of Evaluation Criteria
  - --Table 5.1 page 202 in [Siewiorek and Swarz]

11:30 4.) Design For Reliability
4.1) System Specification (1 Hr)
- -Translating reliability requirements to specs
- -Trade-offs, example of the VAX-11/750, pages 386 to 408 in [Siewiorek and Swarz]

1:30 Guest Speaker: Ann Duncan - Reliability elements in the DEC Software Design Process (1 Hr)
- -Software performance and certification
- -Overview of process
- -Document and code reviews
- -Program design language
- -Methodology
- -Correctness proofs

3:30 Laboratory: Using Prediction Tools to Explore Fault Avoidance in VAX-11/780 (2 Hr)

Day Three - Wednesday

8:30 4.2) Fault Handling Techniques and Their Evaluation (3 Hr)
- -For example, Table 3.7 (page 92) and Table 3.8 (page 95) in [Siewiorek and Swarz]

11:30 5.) Verification Testing Theory
5.1) Overview of Testing Theory (1 Hr)

1:30 Guest Speaker: John Ewalt - Design for Testability at DEC

2:30 Laboratory: Using Prediction Tools for Modeling Redundant Systems (2 Hr)

Day Four - Thursday

8:30 5.2) Design for Testability and Maintainability (1 Hr)
- -pages 191 to 200 in [Siewiorek and Swarz]
- -SPEAR

10:00 Guest Speaker : Bill Bruckert - RAMP in the VAX8600 - A Case Study

11:00 Guest Speaker : Dave Clark - Vax 8600 Elog /Field Data Analysis

1:00 Laboratory: Class Project - Designing a VAX Cluster for Real Time Transaction Processing with 2 minute downtime per year and at most 0.01% transaction re-entry rate.

Day Five - Friday

8:30 Guest Speaker: Kembo Zunzanyika - The DEC Qualification Process

10:00 6.) Commerical Architectures (2 Hr)
- -Taxonomy
- -Selected Examples from Competitors
  - --IBM
  - --Sperry Univac
  - --Bell Systems
  - --Tandem
  - --Stratus
- -A Reliable Design Methodology and Example
  - --The Intel 432

1:30 Laboratory: Finish Class Project (2 Hr)

3:30 Present and Discuss Projects (1 Hr)

Course Feedback

Course is based on "The Theory and Practice of Reliable System Design", by D. P. Siewiorek and R. S. Swarz, Digital Press, 1982.

1.) Attributes of Reliable Computing and Their Definitions

Table 1–1. Levels of abstraction for digital computers.

Level	*Sublevel*	*Components*
PMS		Processors
		Memories
		Switches
		Controllers
		Transducers
		Data operators
		Links
Program	High-level language	Software
	ISP	Memory state
		Processor state
		Effective address calculation
		Instruction decode
		Instruction execution
Logic	Register transfer	Data paths
		Registers
		Data operators
		Control
		Hardwired
		Sequential logic machines
		Microprogramming
		Microsequencer
		Microstore
	Switching circuit	Sequential
		Flip-flops
		Latches
		Delays
		Combinatorial
		Gates
		Encoders/Decoders
		Data operators
Circuit		Resistors
		Capacitors
		Inductors
		Power sources
		Diodes
		Transistors

Stages in the development of a system

Stage	Error Sources	Error Detection Techniques
Specification and design	Algorithm design Formal specification	Simulation Consistency checks
Prototype	Algorithm design Wiring and assembly Timing Component Failure	Stimulus/Response Testing
Manufacture	Wiring and Assembly Component Failure	System Testing Diagnostics
Installation	Assembly Component Failure	System Testing Diagnostics
Field Operation	Component Failure Operator Errors Environmental Factors	Diagnostics

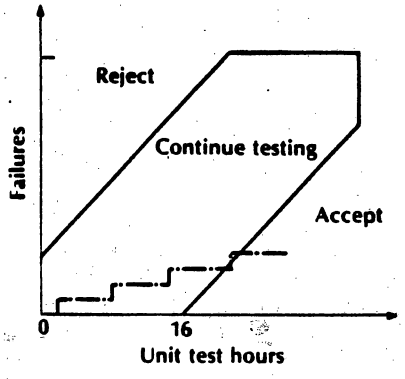


Figure 1–2. Reliability Demonstration Chart for monitoring the progress of a Design Maturity Test.

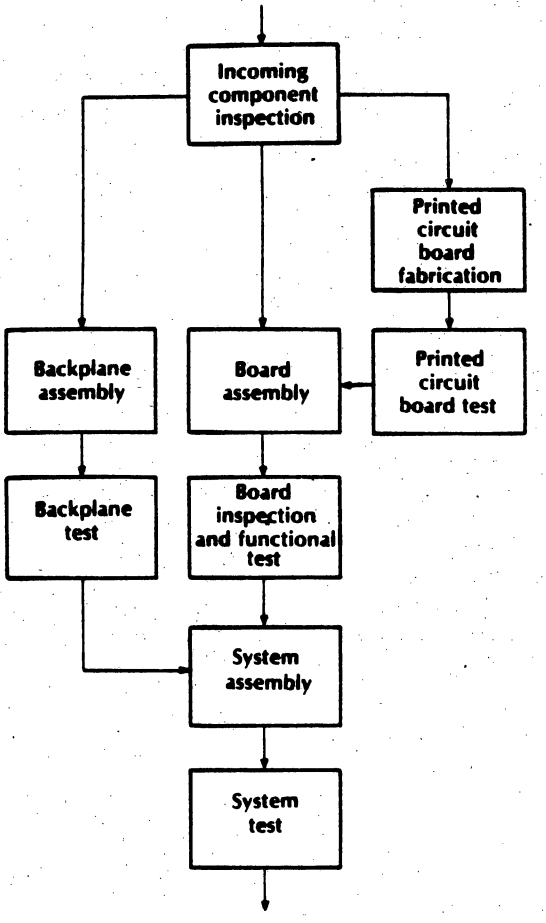


Figure 1–3. Typical steps in the manufacture of a digital system. (© 1979 IEEE.)

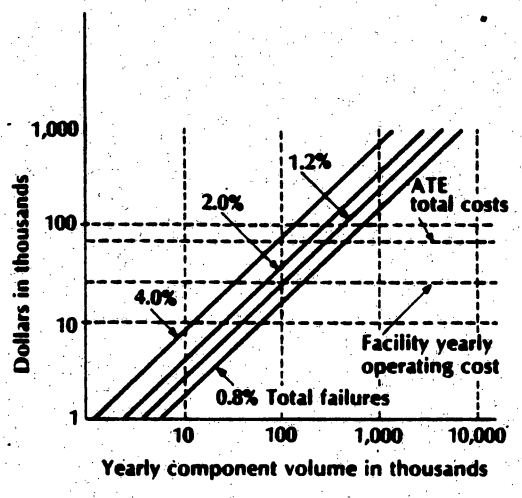


Figure 1–6. Savings from screening and testing as a function of defective component rate and annual device volume.

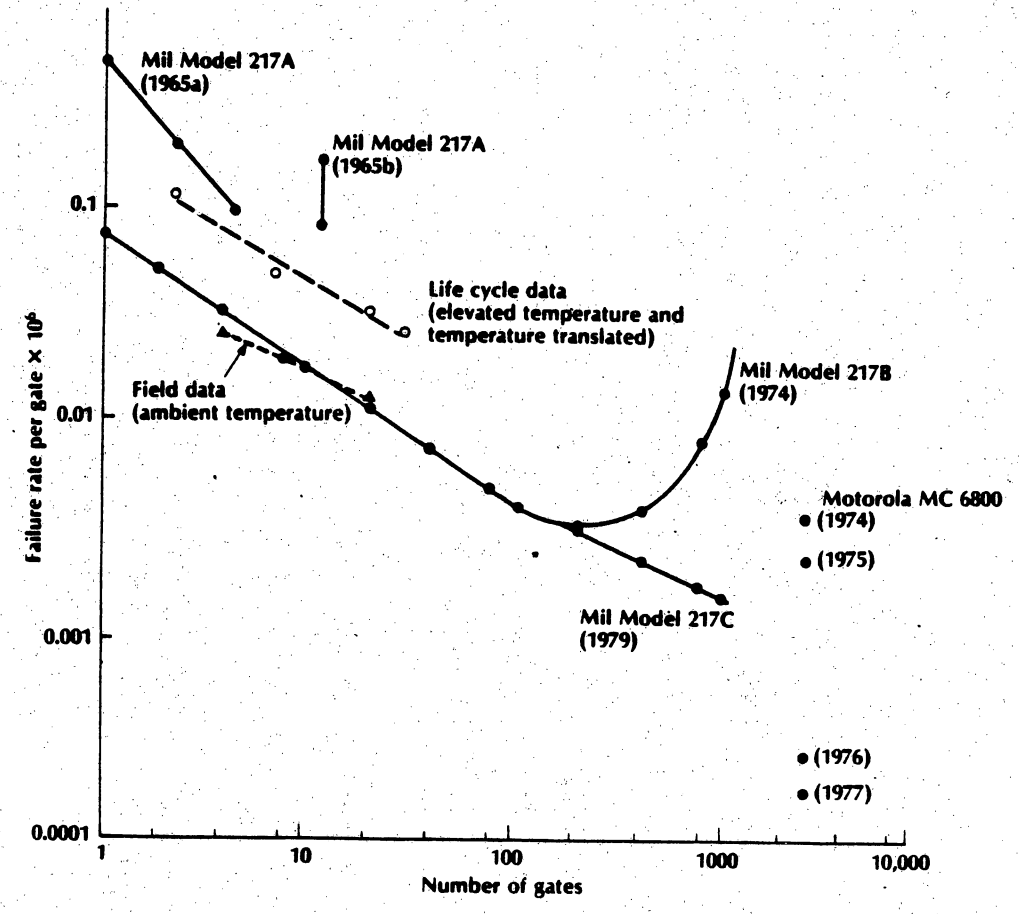


Figure 1–5. Failure rate per gate as a function of chip complexity for bipolar technology.

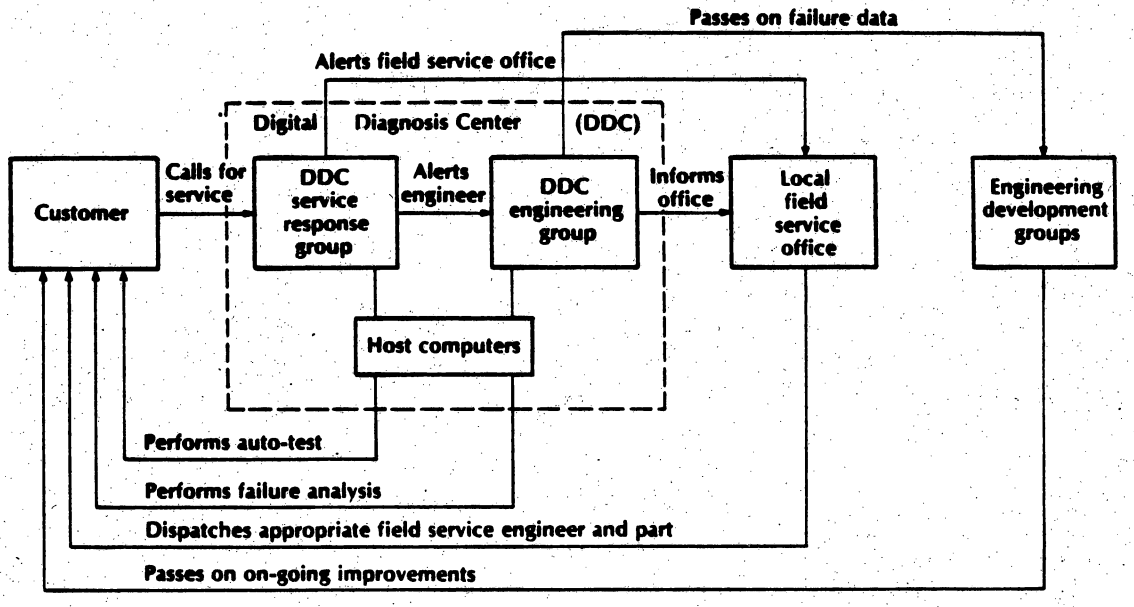


Figure 1–7. Overview of DEC's Remote Diagnosis Network.

Cost of Ownership

$$C = I + \sum_{i=1}^{n} \frac{S_i P_i}{(1+D)^i}$$

S_i - cost of one call

P_i - expected number of failures/year

D - Discount rate

For $S_i = \$300$

$n = 5$

$D = .2$

$$C = I + 7.86\lambda$$

λ in fpmh

Table 1–3. Parameters a designer can control, their impact on system design goals, and typical techniques used to achieve these goals.

Designable Parameters	*Goals*	*Example Techniques*
	Hard Failures	
MTTF	Tolerate	Replication
MTTR	Isolate	Detection
	Transient Faults	
MTTC	Tolerate	Detection/Retry

RELIABILITY: SURVIVAL PROBABILITY

When repair is costly or a critical function

AVAILABILITY: THE FRACTION OF TIME A SYSTEM MEETS ITS SPECIFICATION

When service can be delayed or denied

Try to find out how do systems fail

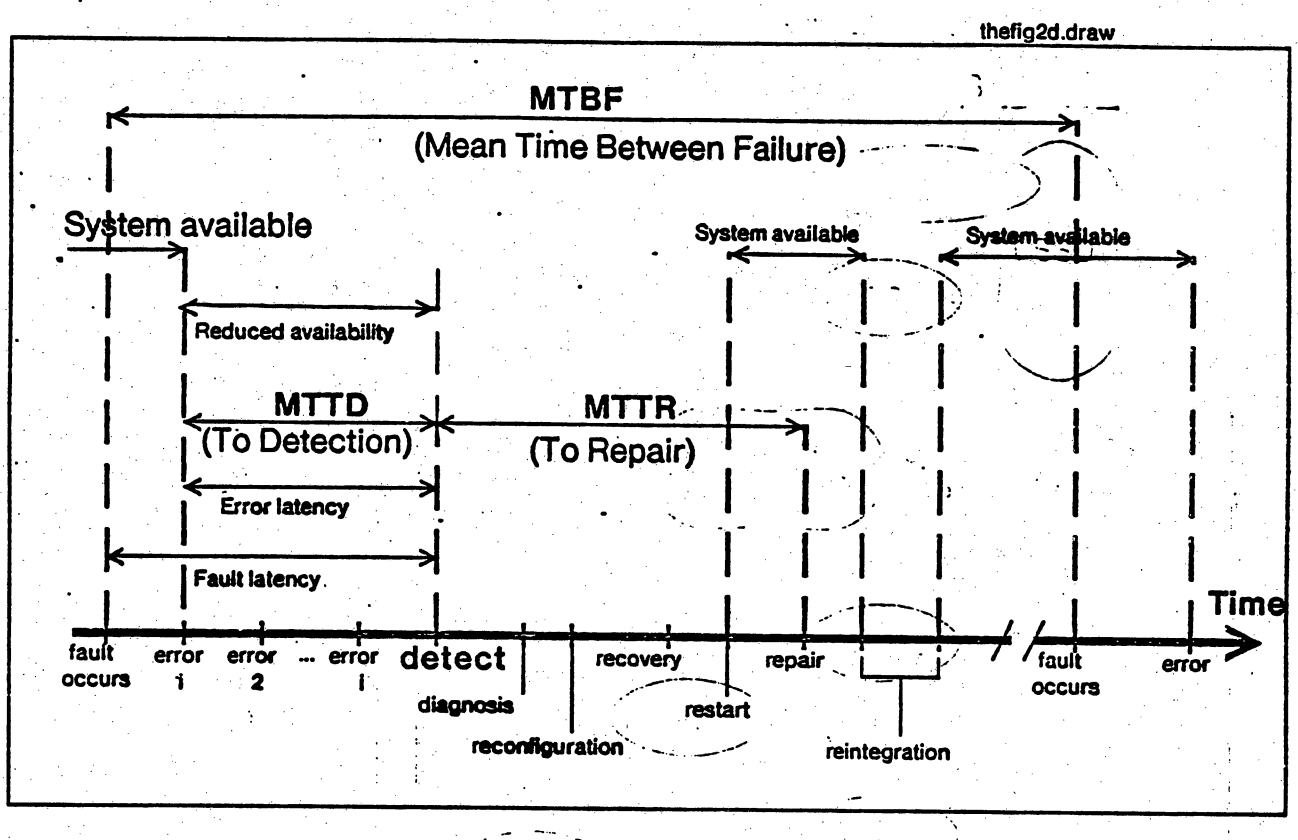
1.10

Redundancy: Extra hardware, software, time

Basic Functions:

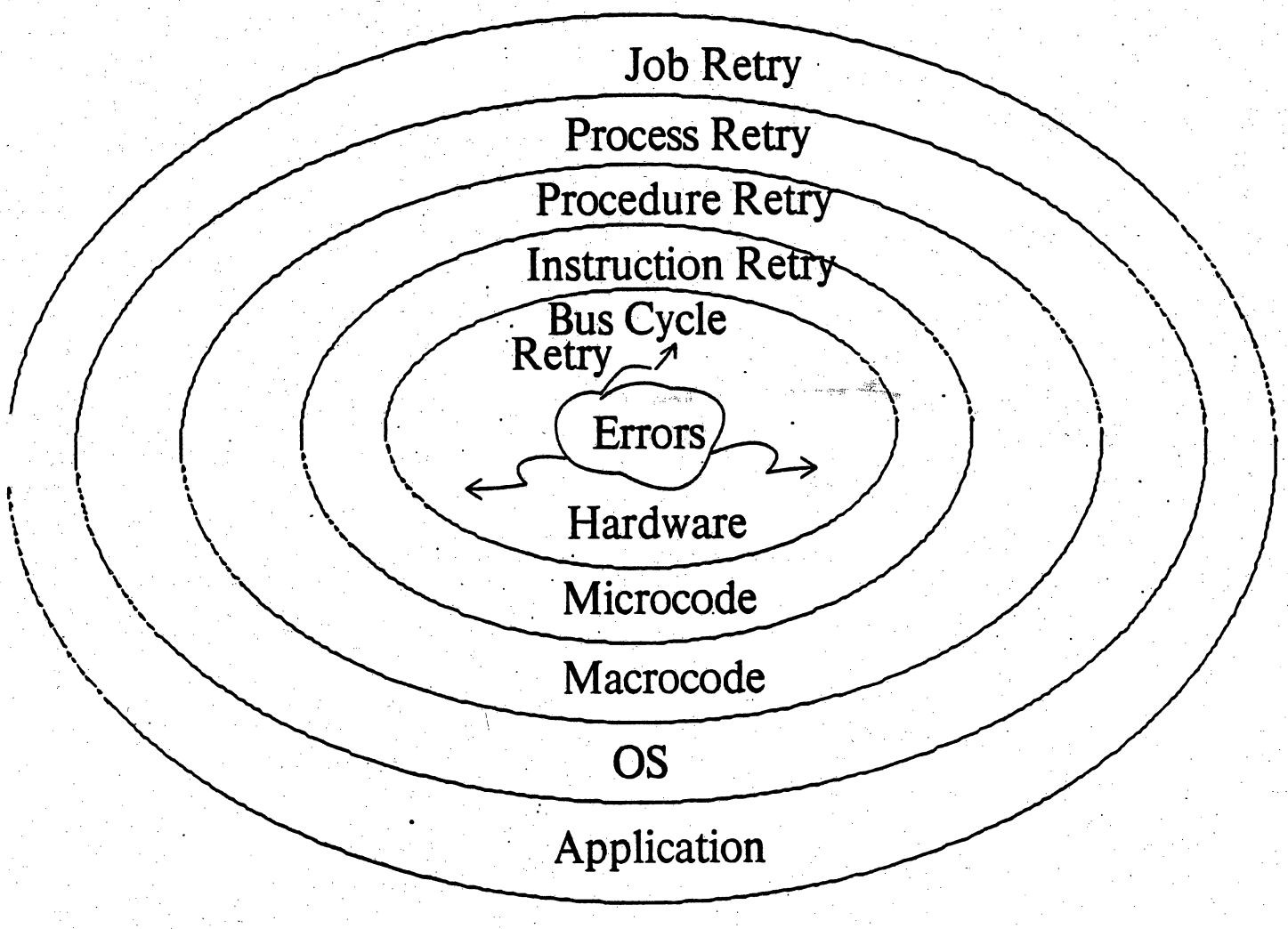
- Fault Confinement
- Fault Detection
- Fault Masking
- Retry
- Diagnosis
- Reconfiguration
- Recovery
- Restart
- Repair
- Reintegration

Error History



Levels in a System

. Detect errors early - limit damage
. Correct at lowest possible level - limit recovery complexity



The further out the error propagates, the more state that is involved and the more diverse error manifestations, and the more complex the error recovery.

Hardware Error Tolerance Advantages

- Simplifies recovery for SW and user applications
- Saves time
- provides transparency to the user
- decreases MTTR
- increases MTTF, MTTE, MTT
- error detection logic can help isolate design errors so that future implementations are even more reliable

2.) Why Design for Reliability

WHY RELIABILITY?

Three of the driving factors:

Critical Applications

- computer outage or error can cause loss of money, time, life.
- no longer just in aerospace, but in more mundane applications.

Increasing System Complexity

- more components ⟶ more likelihood of failure (counter: increased rel. of ↑ LSI).
- lower signal/noise ratios in ↑ LSI speed ⟶ more likelihood of transient errors.
- diagnosis more difficult, downtime is longer, repair costs ↑ , increased inventory costs, too.

Relative Cost is Less

AVAILABILITY EXAMPLE

Availability
0.991

90 minutes downtime per week

Reservation system -- $36,000 / minute down

$3.24 million per week

.1% availability = 10 minutes

= $360,000.00

ANATOMY OF A TYPICAL OUTAGE

0 Minutes: Problem occurs

3 Minutes: Operator decides problem requires dump/restart

8 Minutes: Operator completes dump

12 Minutes: OS restart complete; begin DB/DC restart

17 Minutes: DB restart complete (assume no tape handling)

30 Minutes: Network restart continuing

40 Minutes: Network restart continuing

50 Minutes: Network restart continuing

60 Minutes: Network restart continuing

70 Minutes: DC restart complete; begin user restart

80 Minutes: User restart continuing

90 Minutes: User restart complete

A simple failure mushrooms into a 90 minute outage.

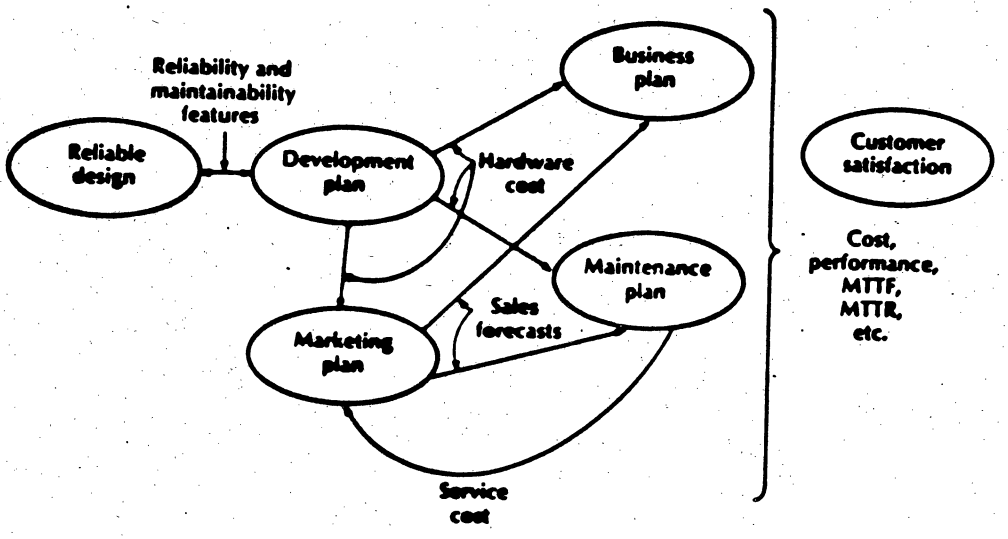
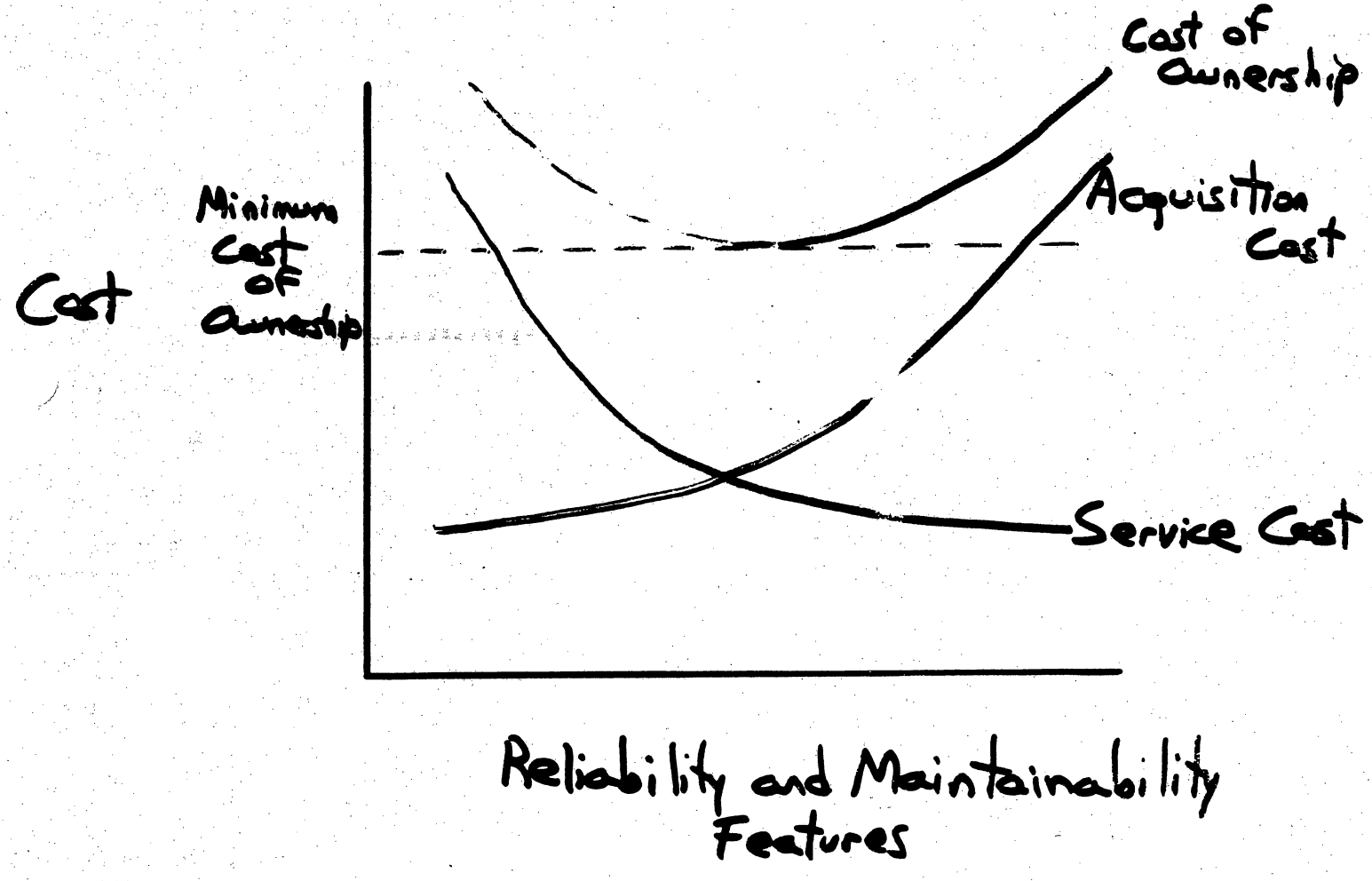


Figure 1–8 Major activities in the design and marketing of a computer system.


Cost of ownership
Acquisition Cost
Service Cost
Minimum cost of ownership
Cost
Reliability and Maintainability Features

3.1) Faults and Their Manifestations

CAUSE-EFFECT SEQUENCE

FAILURE - PHYSICAL CHANGE IN HARDWARE

FAULT - DEVIATION OF LOGIC FUNCTION FROM DESIGN VALUE

HARD, TRANSIENT

ERROR - MANIFESTATION OF A FAULT BY INCORRECT VALUE

FAULT CLASSIFICATION

DURATION - TRANSIENT — design errors, environment

Intermittent } Repair by replacement

PERMANENT }

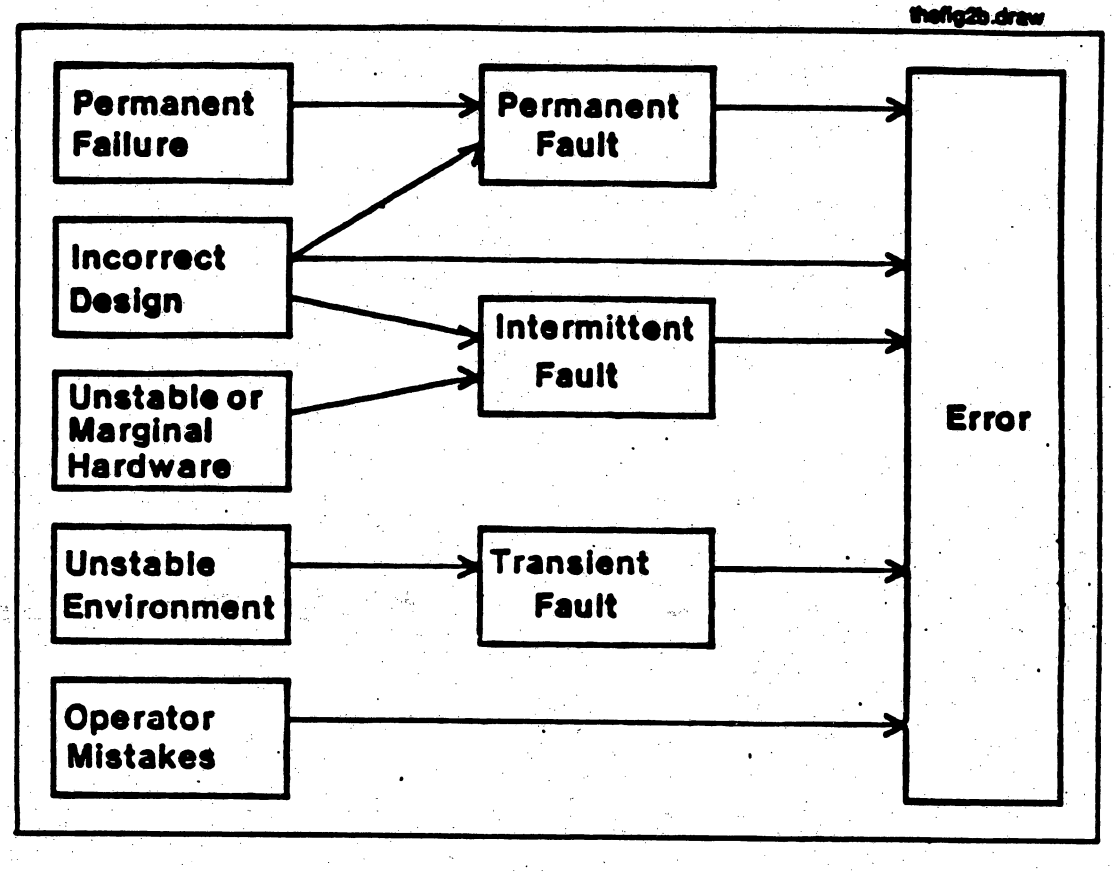
EXTENT - LOCAL (INDEPENDENT)

DISTRIBUTED (RELATED)

VALUE - DETERMINATE (STUCK AT X)

INDETERMINATE (VARIABLE)

Definitions:
Sources of Errors



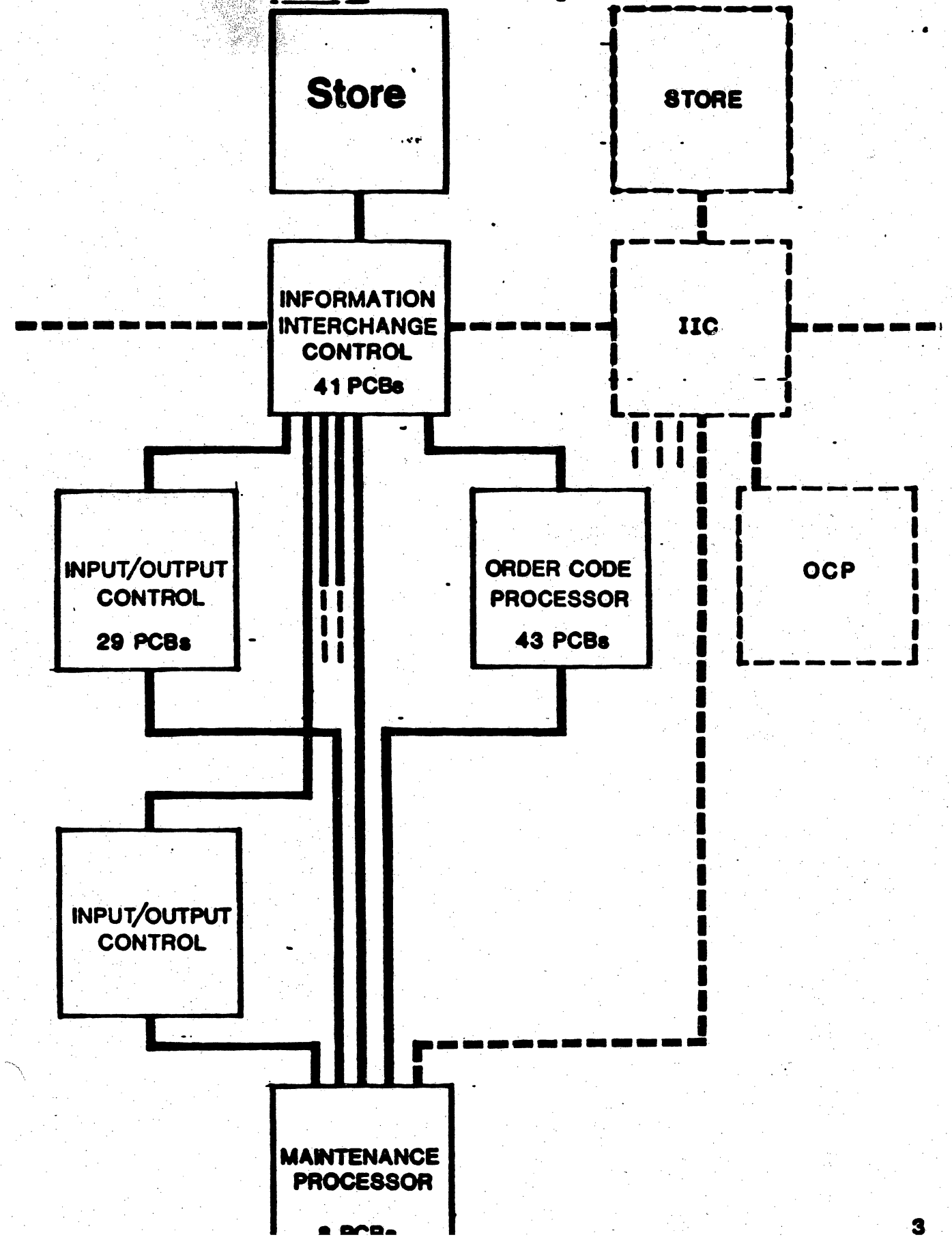
Design environment

Design fault data analysed for:-

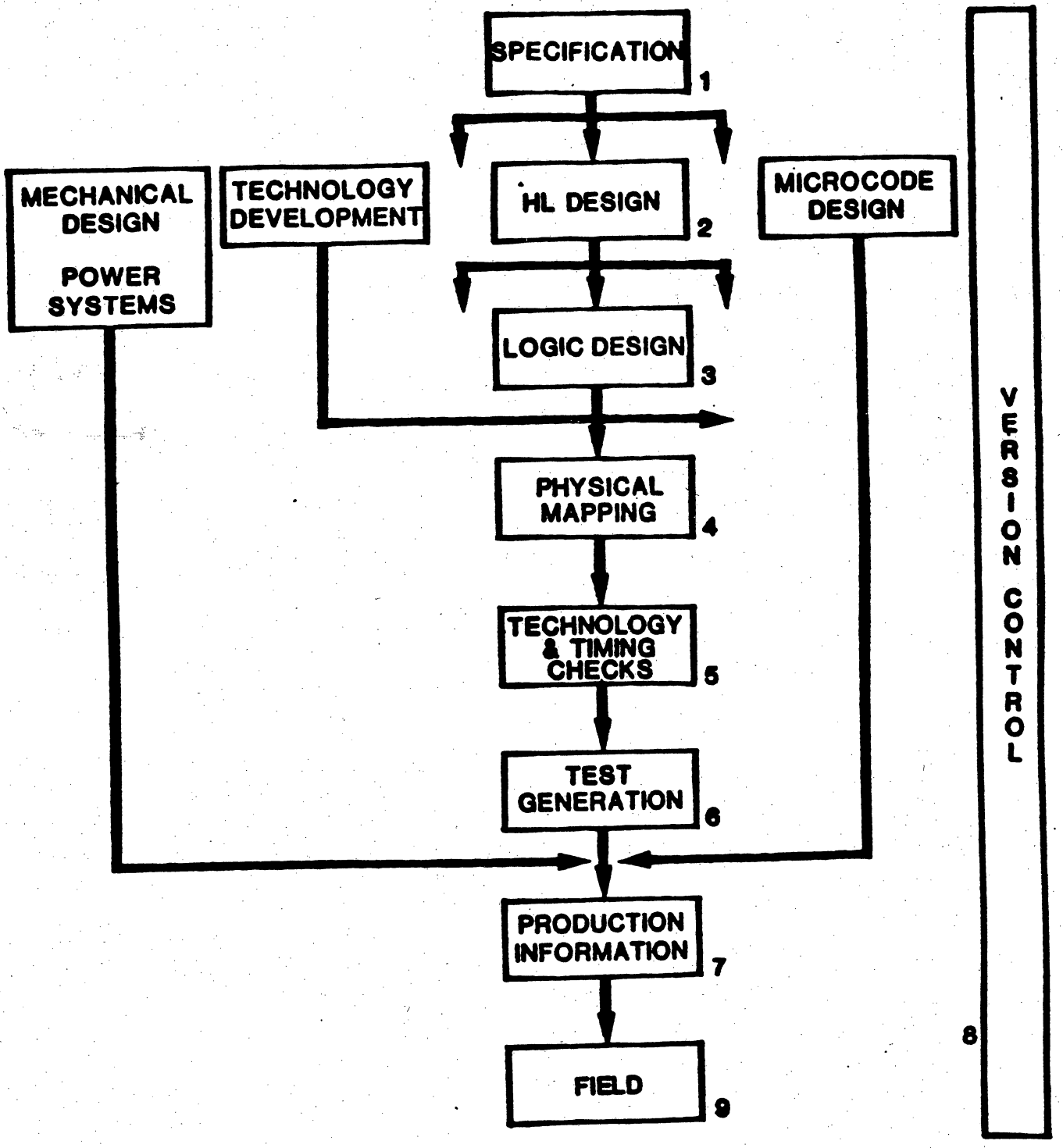
* Mid range mainframe
* Schottky TTL implementaion
* Incremental development
* Extensive design automation
* Design modularity
* Separate module design projects

Circa early 1980's

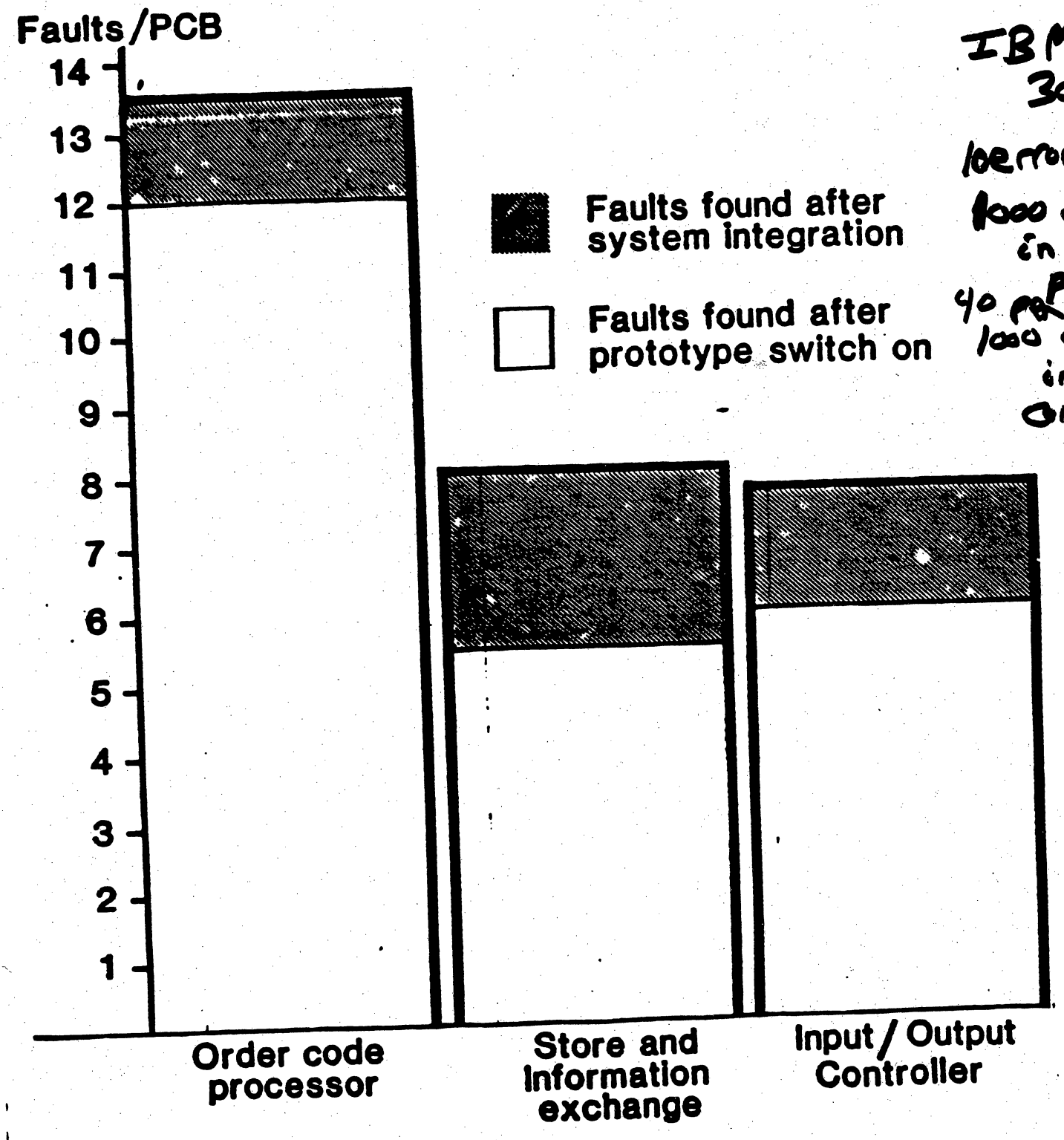
Structure of product



Typical development route



When faults are found



IBM 3081

10 error per 1000 circui in data path

40 per 1000 circuit in control

Fault classification

Specification faults

- Errors
- Ambiguities
- Ommissions

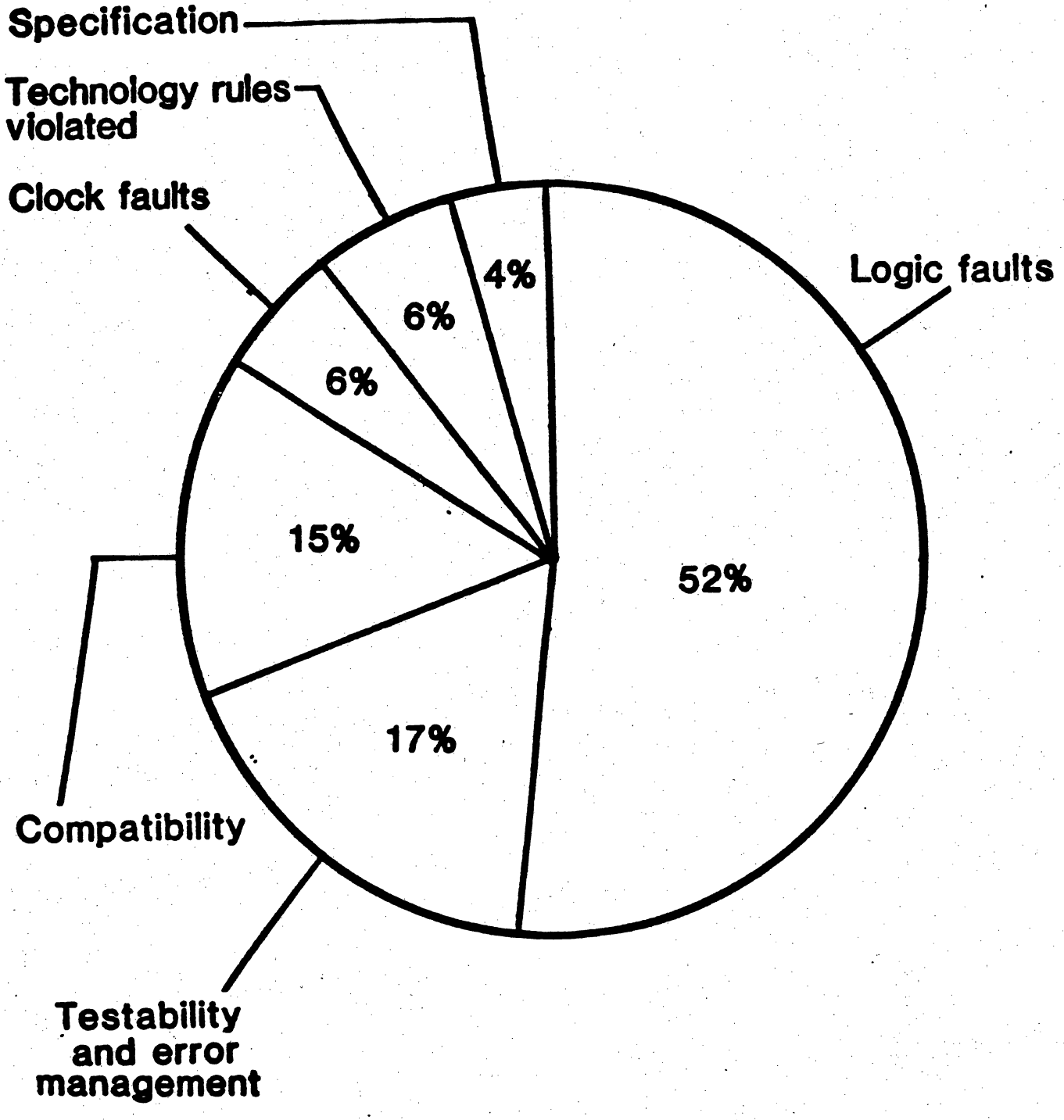
Environment faults

- Technology rules violated
- Noise
- Design Automation errors

Realisation faults

- Simple logic error
- Concurrency problem
- Time out
- Initialisation
- Clock fault
- Performance/Size problem
- Error management fault
- Testability/Maintainability
- Compatability

Breakdown of types of faults found BEFORE system integration



Breakdown of types of faults found AFTER system integration

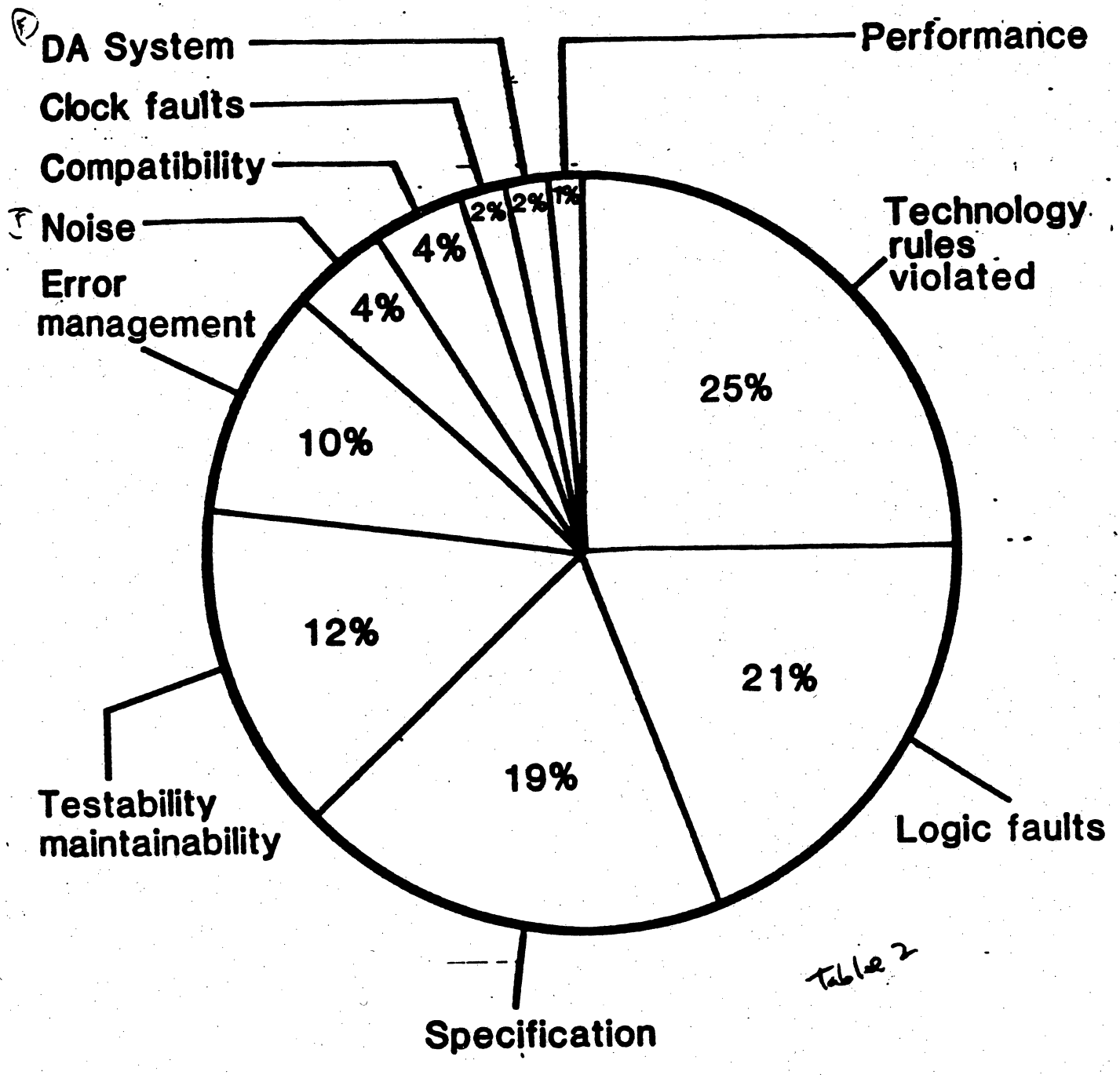
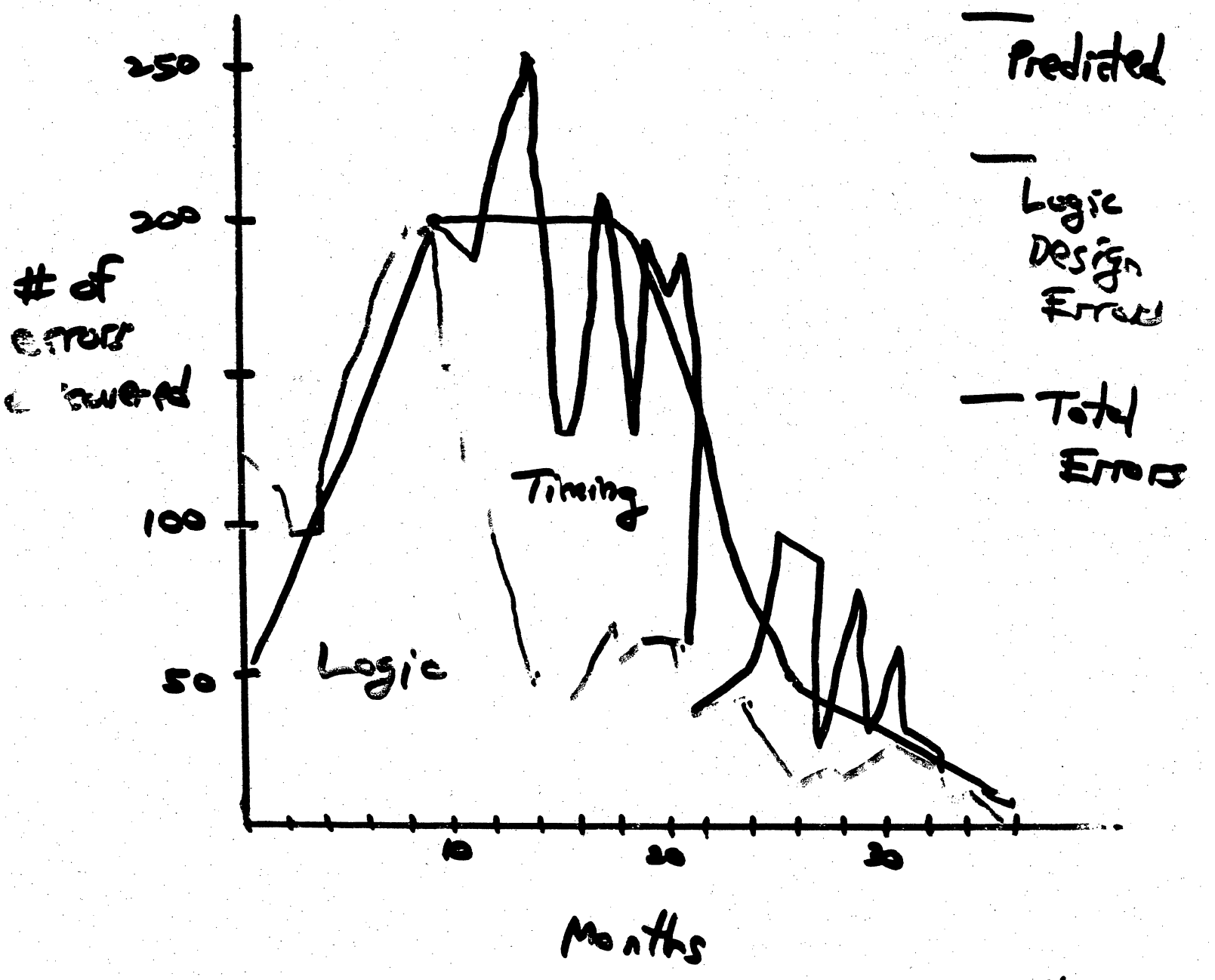


Table 2

History

95% error free

Coverage
- 90% single
- 70% double
- 40% triple decision paths



10 errors/1000 circuits - data path

40 errors/1000 circuits - control

[illegible] logic 100% timing is verifiable

Summary of Interrupts:

Intrpt No.	Assignment/ Function	Mode/ Effect
8	unassigned	
9	unassigned	
A	Arithmetic Overflow[1]	USER/Stalls system PRIV/No effect
B	IPC interrupt	
C	Interval timer	
D	Write Protection Violation[1]	USER/Stalls system PRIV/Write protection ignored
E	Page Fault	Still being evaluated
F	Test Adaptor[4]	
10	Halt Instruction Execution[1]	
11	Illegal Opcode[1]	USER/stalls system PRIV/ignored
12	Stack Overflow[1]	USER/stalls system PRIV/ignored
13	Non-local Search Fault[1,2,4]	
14	Privileged instr Fault[1,2]	USER/stall system
15	pmcall fault[1,3]	USER/No Privileged mode routines
16	unassigned	
17	unassigned	
•	Divide exception[5]	USER or PRIV/Stalls system.

Many of the interrupts do not have a proper interrupt handler:

- Arithmetic Overflow
- Write Protection Violation
- Illegal Opcode
- Stack Overflow
- Non-local Search Fault
- Privileged Instruction Violation
- Privileged Mode Call Fault

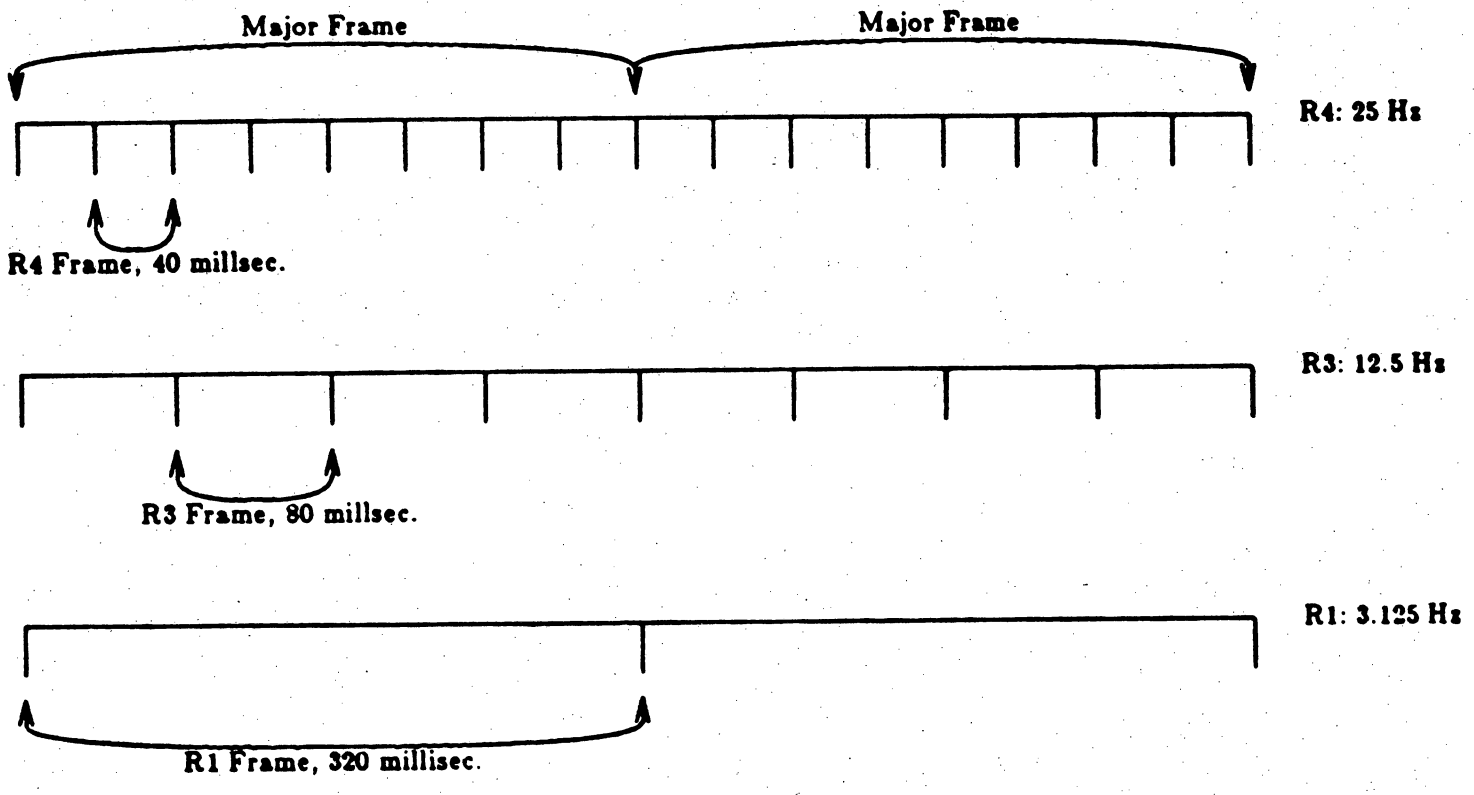
Use in USER Mode crashes machine.

In addition, there is no interrupt vector for Divide Exception.

To avoid stalling the system due to interrupts, all application software in run in PRIVILEGED mode (interrupts ignored).

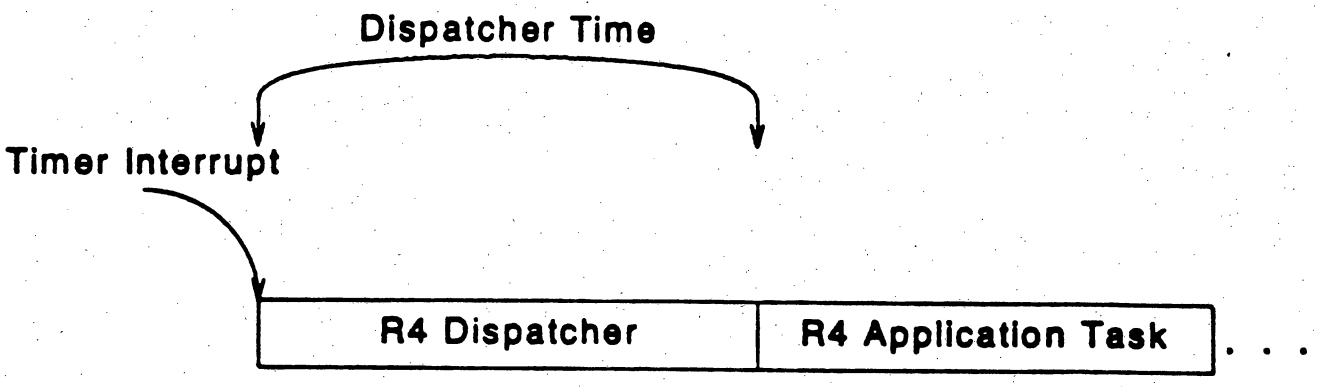
FTMP Task Frame Structure.

- Task: A single "program" or thread of execution.
- Frame: The period of time which a task is to execute.
- Iteration Rate: The rate or frequency which tasks are executed on FTMP.



Dispatcher Execution Time:

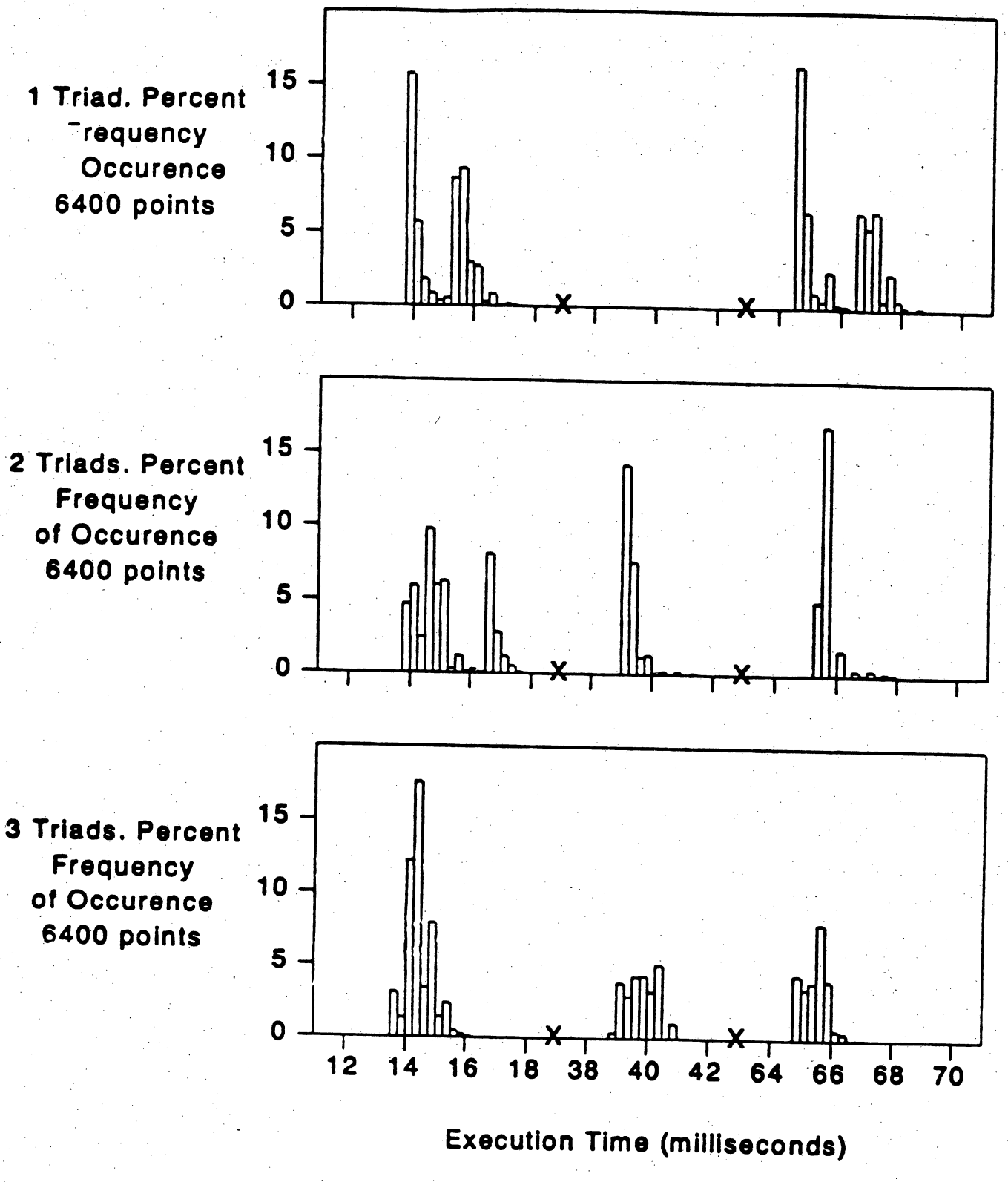
Frame Start to First Application Task.



Dispatcher performs these functions:

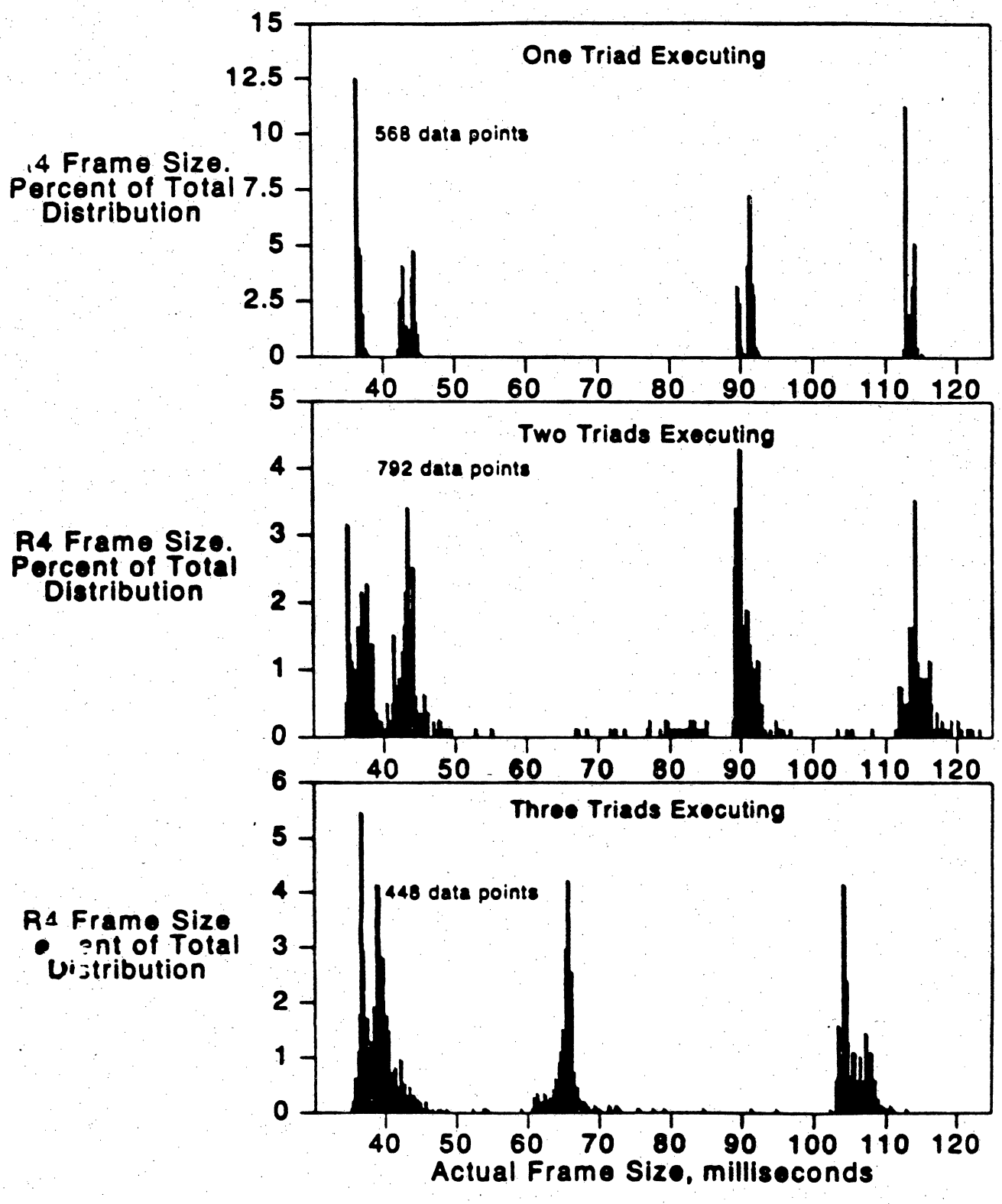
- Schedules the start of R3 and R1 frames,
- Carrys out Sensor and Display Terminal I/O,
- Executes reconfiguration commands, and
- Does necessary task I/O.

Dispatcher Execution Time:

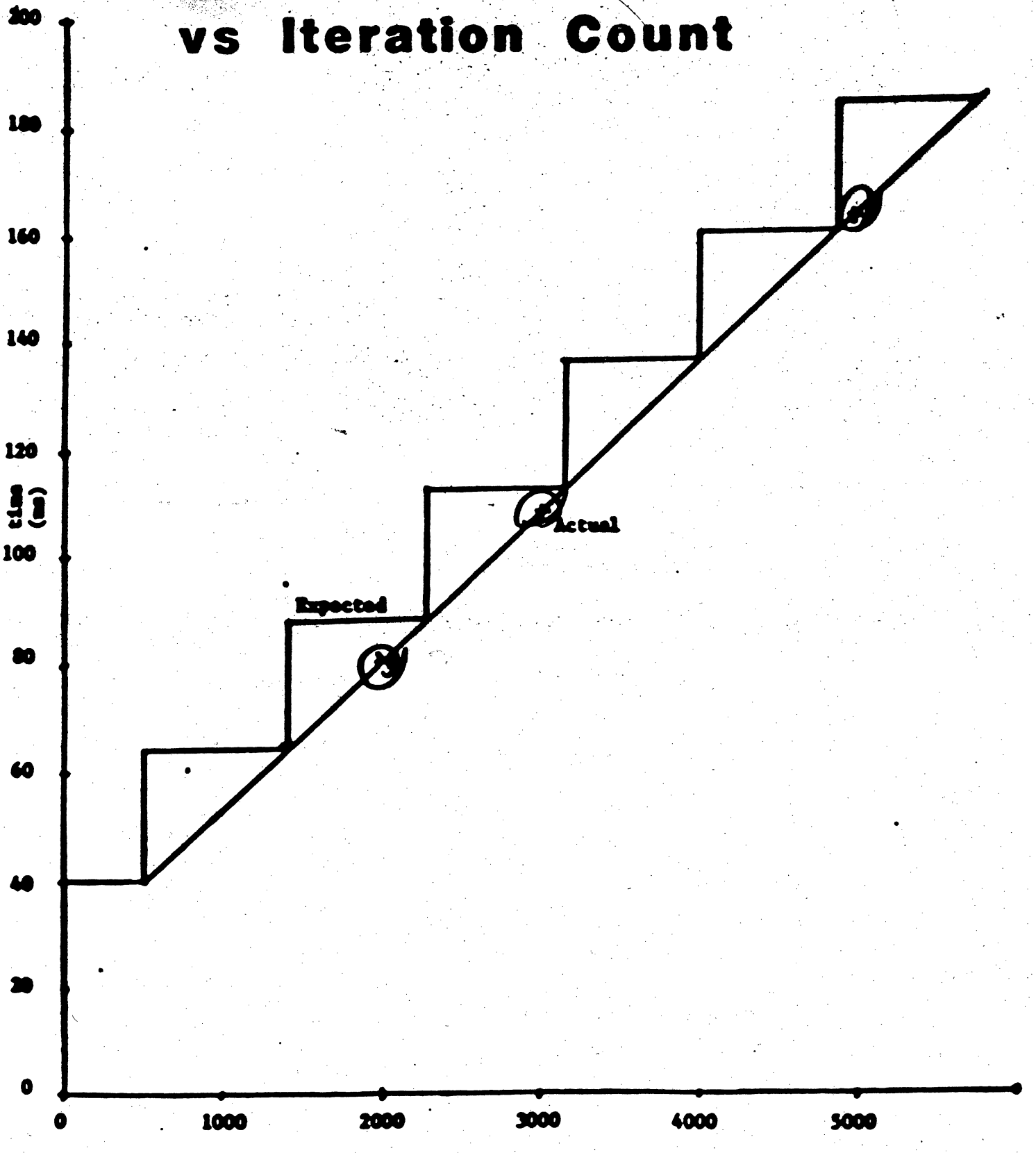


R4 Frame Sizes:

The true time between tasks in consecutive frames.



Frame Size vs Iteration Count



QUESTIONS ABOUT FAULTS

- How often do they occur?
- How do they manifest themselves?
- What is the shape of the function?
- What are the parameters of the function?
- What are the meanings of related events?

Trend analysis

System	Mean Time to Restart	Mean # of Instr. to Restart	% Hw	% Sw	% Unknown
B5500	14.7	2.6×10^{10}	39.3	8.1	52.6
Chi/05 (Univac 1108)	17	6.7×10^{10}	45	55	-
Dual 370/165	8.86	2.8×10^{11}	65	32	3
SLAC 2-370/168 1-360/91	20.2	2.3×10^{11}	73.3	21.6	5.1
PDP-10	10	4.3×10^{10}	-	-	-
Univac-large	-	-	51	42	7
Univac-Med.	-	-	57	41	2
Univac-Small	-	-	88	9	3

IBM 3081

% Sw	% Hw	Hw Transient/Permanent
26	74	16.9:1

40% of Sw failures Hw related

Software Errors	MVS/SP	MVS/XA
MTTE (hours)		
Critical Jobs	25	41
All Jobs	6.5	6.1
MTEH (seconds)	41	86

Distribution of SW Errors

Test Programs	26.8%
System Programs	24.6
User Programs	11.3
Interactive System	12
Other	8.8
Unknown	14.5

SW Errors

	Detection Mechanism in SW (%)	Recovered %
Deadlock	95	100
I/O & Data Management	53	100
Storage Management	100	33
Programming Exceptions	32	0
Storage Exceptions	16	12
All	-	24

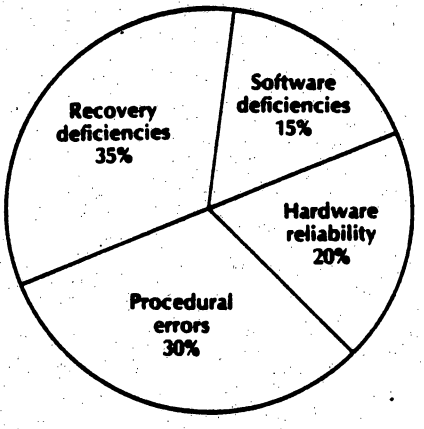


Figure -1. System outage allocation.

TABLE 21

INFLIGHT BIT TEST PROCEDURES

COMPONENT	METHOD OF FAILURE DETECTION
1) CPU	1) CPU Self-Test
2) Real Time Clock	2) Watchdog Timer
3) Program Memory	3) Parity & Memory Sum Test
4) Scratchpad Memory	4) Parity Redundant Memory Addressing and Bit Pattern Tests
5) DA, AD Converters Sample and Hold Circuits Valve Drive Amplifiers	5) Wrap-arounds
6) Input Multiplexers	6) Bias Inputs, Patterns of Faults
7) Discrete Inputs, Outputs	7) Wrap-arounds Patterns of Faults
8a) I/O Controller Sequencer	8a) Patterns of Faults
8b) I/O Controller File Memory	8b) Parity Tests Memory Sum Test
9) Intercomputer Data Links	9) Not detectable
10) AC, DC Inputs	10) Reasonableness and Reconfiguration
11) PC Boards and Connectors	11) Detection dependent on affected devices
12) Power Supply	12) Level Detectors

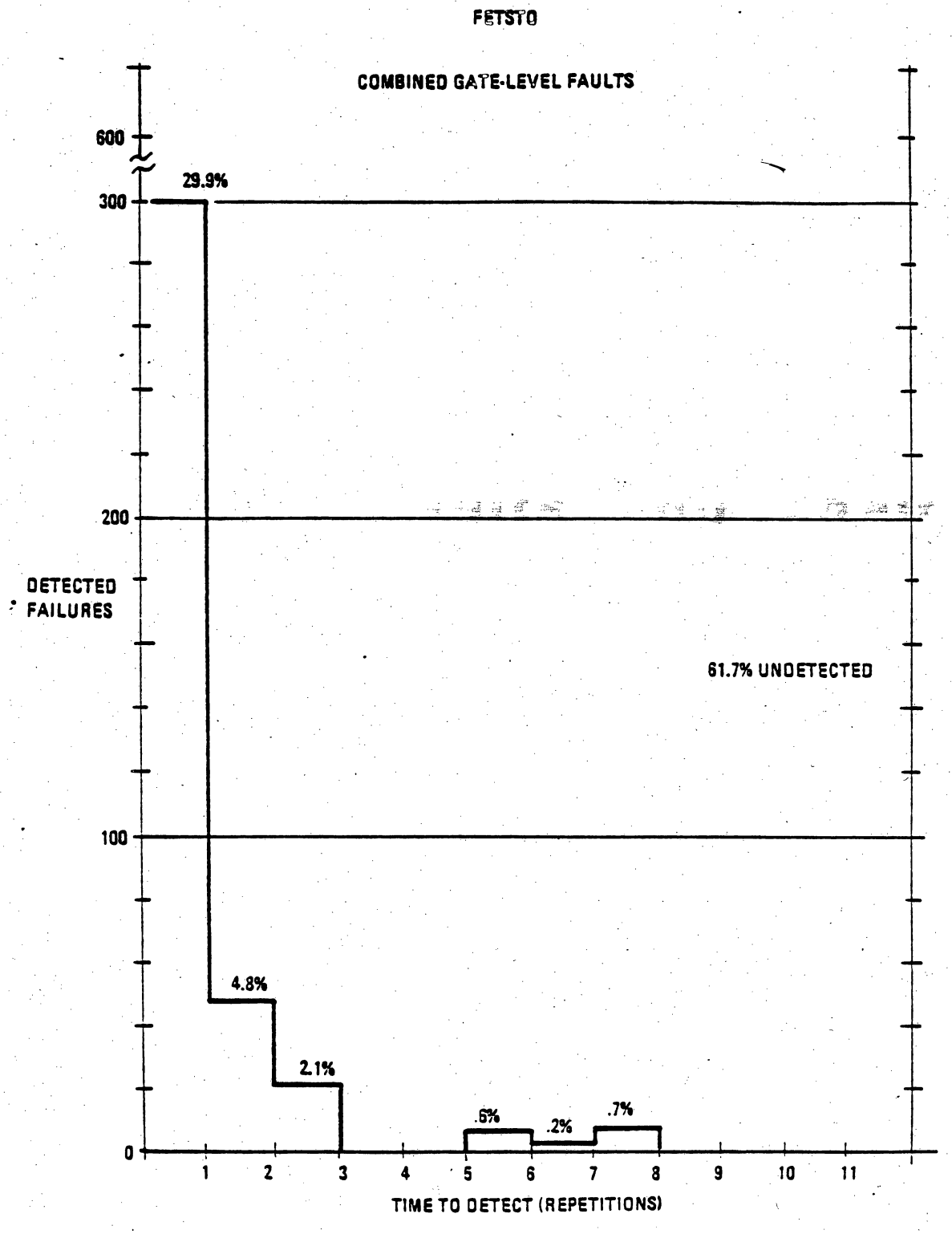

FETSTO
COMBINED GATE-LEVEL FAULTS
600
300
200
100
0
29.9%
61.7% UNDETECTED
4.8%
2.1%
.6%
.2%
.7%
DETECTED FAILURES
1
2
3
4
5
6
7
8
9
10
11
TIME TO DETECT (REPETITIONS)

FIGURE 2a

FETSTO

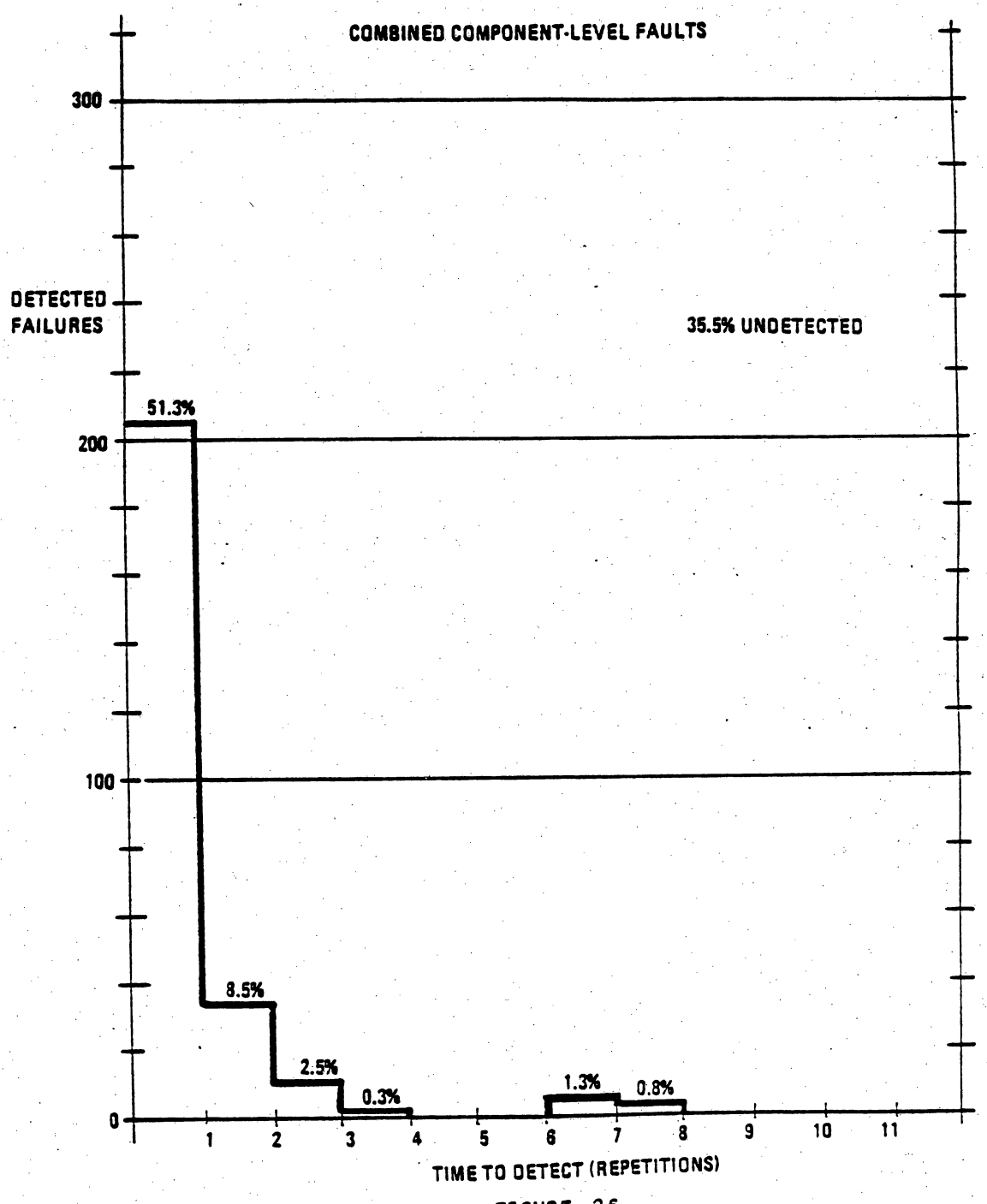


FIGURE 2f

System Failure Mode	Probability		MTBF in years
Administration	42%		31 years
Maintenance:	25%		
Operations	9%	(?)	
Configuration	8%		
Software	25%		50 years
Application	4%	(?)	
Vendor	21%		
Hardware	18%		73 years
Central	1%		
Disc	7%		
Tape	2%		
Comm Controllers	6%		
Power supply	2%		
Environment	14%		87 years
Power	9%	(?)	
Communications	3%		
Facilities	2%		
Unknown	3%		
Total	103%		11 years

Table 1. Contributors to Tandem System outages reported to the vendor. As explained in the text, infant failures (30%) are eliminated from this sample set. Items marked by "?" are probably under-reported because the customer does not generally complain to the vendor about them. Power outages below 4 hours are tolerated by the NonStop system and hence are under-reported. We estimate 50% total under-reporting.

Table 2–5. Die-related defect summary: SSI, MSI, LSI CMOS.

General Defect Classification	*No. Malfunctions*	*Relative Percent*
Surface	65	38%
Bulk	12	7
Oxide	54	32
Diffusion	10	6
Metalization	14	8
Input/output circuit	15	9

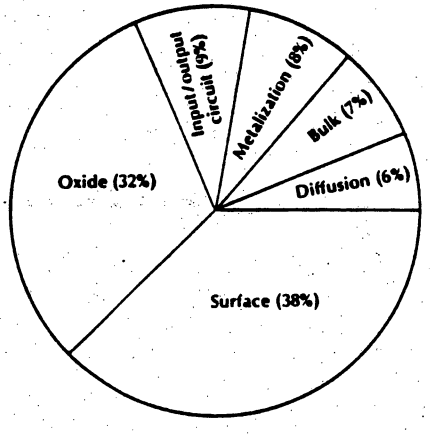


Table 2–6. Die-related failure modes: SSI, MSI, LSI standard TTL.

Failure Classification	SSI STD TTL		MSI STD TTL		LSI STD TTL	
	No. Devices	%	No. Devices	%	No. Devices	%
Surface defects	51	20%	10	11%	5	8%
Contamination	29	11	5	6	2	3
Foreign material/ stray particles	7	3	4	5	1	2
Inversion/channeling	11	4	1	1	2	3
Surface leakage	4	2				
Bulk defects	24	9	5	6	0	0
Crystal imperfections	2	1	3	3		
Cracked, chipped die	22	8	2	2		
Oxide defects	27	10	10	11	22	33
Gate oxide pinholes						
Field oxide pinholes	8	3	6	7	12	18
Oxide fault	19	7	3	3	3	4
Oxide short/ breakdown					5	8
Passivation defect			1	0	2	3
Diffusion defects	19	7	13	15	0	0
Diffusion anomaly	4	2	9	10		
Diffusion spike/ piped junction	3	1	2	2		
Isolation defect	1	0				
Mask fault	11	4	2	2		
Metalization defects	136	52	38	43	39	59
Open at oxide step						
Open at contact window	85	33	4	5		
Open/not specified	13	5	10	11	11	17
Short/interlayer metal	8	3	7	8	9	13
Short/not specified	22	8	15	17	17	26
Pitted/corroded	5	2				
Smeared/scratched	3	1	2	2	2	3
Electromigration						
Input/output circuit defects	5	2	12	14	0	0
Excessive input leakage	1	0	6	7		
Input circuit short	2	1	4	5		
Excessive output leakage			1	1		
Output circuit short	2	1	1	1		
Total	262		88		66	

Table 2–7. Die-related defect summary: SSI, MSI, LSI standard TTL.

General Defect Classification	*No. Malfunctions*	*Relative Percent*
Surface	66	16%
Bulk	29	7
Oxide	59	14
Diffusion	32	8
Metalization	213	51
Input/output circuit	17	4

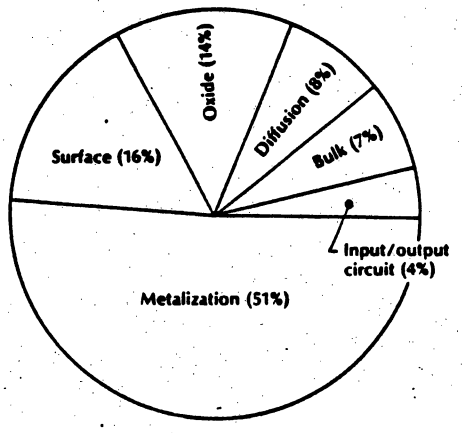


Table 2–8. Screening test summary.

Screening Tests	*Substrate Bonding Defects*	*Substrate Thin Film Defects*	*Bulk Silicon Defects*	*Silicon Surface Defects*	*Contamination or Corrosion*	*Wire Bonding Defects*	*Die Metallization Defects*	*Electrical Stability*	*Resistor and Chip Defects*	*Hermetic Seal Defects*	*Package Defects*	*External Lead Defects*	*Cracked Die or Substrates*
Wafer probe		x	x				x	x	x				
Wafer inspection				x	x		x					x	
Precap inspection	x	x		x	x	x	x					x	
Stabilization bake			x	x	x	x	x	x					
Thermal cycling	x	x		x		x	x		x	x	x	x	
Thermal shock	x	x		x		x	x		x	x	x	x	
Hermeticity										x	x		
Centrifuge	x				x	x					x		
Mechanical shock	x				x	x					x	x	
Vibration	x				x	x					x		
Burn-in		x	x	x	x	x	x	x					
Radiographic	x				x	x				x		x	
External visual										x	x	x	
Scanning Electron Microscope		x		x	x		x					x	

IFA Results

	Fault Types	Number	Percentage
1.	Line Stuck At Faults	132	28%
2.	Transistor Stuck At Faults	70	15%
3.	Floating Line Faults	101	21%
4.	Bridging Faults	144	30%
5.	Miscellaneous Faults	29	6%
	Total	476	100%

Some Observations

- Nonclassical faults are likely
- Not all faults are equally likely
- Only 28% of the observed faults are explicitly Line Stuck At 0/1 Faults.
- At best only 64% (28% + 15% + 21%) of the faults can be modeled by the stuck at fault model.
- Bridging Faults constitute a large percentage (30%) of the faults.

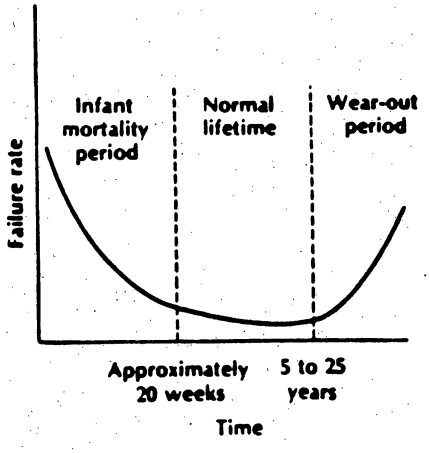


Figure 1–4. Bathtub curve depicting component failure rate as a function of time.

Hazard Function Z(t)

for a known distribution,

$$Z(t) \triangleq \frac{pdf}{1-CDF} = \frac{f(t)}{R(t)}$$

for the type of distributions normally encountered:

$$R(t) = e^{-\int_0^t Z(x)dx}$$

Exponential Distribution (CFR)

$$Z(t) = \lambda$$

Weibull Distribution (IFR, DFR)

$$Z(t) = \alpha\lambda(\lambda t)^{\alpha-1}$$

Poisson Distribution

- Probability of transition from state with N occurrences to state with N + 1 occurrences during time Δt

$$= \beta \Delta t$$

- Occurrences are independent

- Probability of 2 or more occurrences during time Δt is negligible

$$\Longrightarrow Pr\{k \text{ occurrences}\} = \frac{\beta^k e^{-\beta}}{k!}$$

$$E[k] = \sum_{k=\phi}^{\infty} k \frac{\beta^k e^{-\beta}}{k!} = \beta = \text{mean}$$

$$Var[k] = E[k^2] - (E[k])^2 = \beta = \text{variance}$$

Note: $Pr\{\phi \text{ occurrences}\} = e^{-\beta}$

Poisson Distribution (cont.)

Substitute constant failure rate λ for β and evaluate at time t.

=> Pr{k failures in time[0,t]} =

$$\frac{e^{-\lambda t}(\lambda t)^k}{k!}$$

(mean = λt
variance = λt)

DEFINITION

Reliability = R(t) =
Pr{0 failures in time [0,t]}

For constant failure rate λ,
$R(t;\lambda) = e^{-\lambda t}$

Poisson Distribution (cont.)

Substitute hazard function Z(t) for β and evaluate at time t

=> Pr{k failures in time [0,t]} =

$$\frac{e^{-\int_0^t Z(x)dx}\left(\int_0^t Z(x)dx\right)^k}{k!}$$

mean = variance = $\int_0^t Z(x)\,dx$

R(t) = Pr{0 failures in time [0,t]} =

$$e^{-\int_0^t Z(x)dx}$$

[remember prior definition of Z(x)]

Poisson Distribution (cont.)

Substitute $Z(t) = \alpha\lambda(\lambda t)^{\alpha-1}$ for β and evaluate at time t

=> Pr{k failures in time [0,t]} =

$$\frac{e^{-(\lambda t)^{\alpha}}(\lambda t)^{\alpha k}}{k!}$$

mean = variance = $(\lambda t)^{\alpha}$

R(t) = Pr{0 failures in time [0,t]} =

$$e^{-(\lambda t)^{\alpha}}$$

FACTS WE HAVE LEARNED ABOUT TRANSIENTS

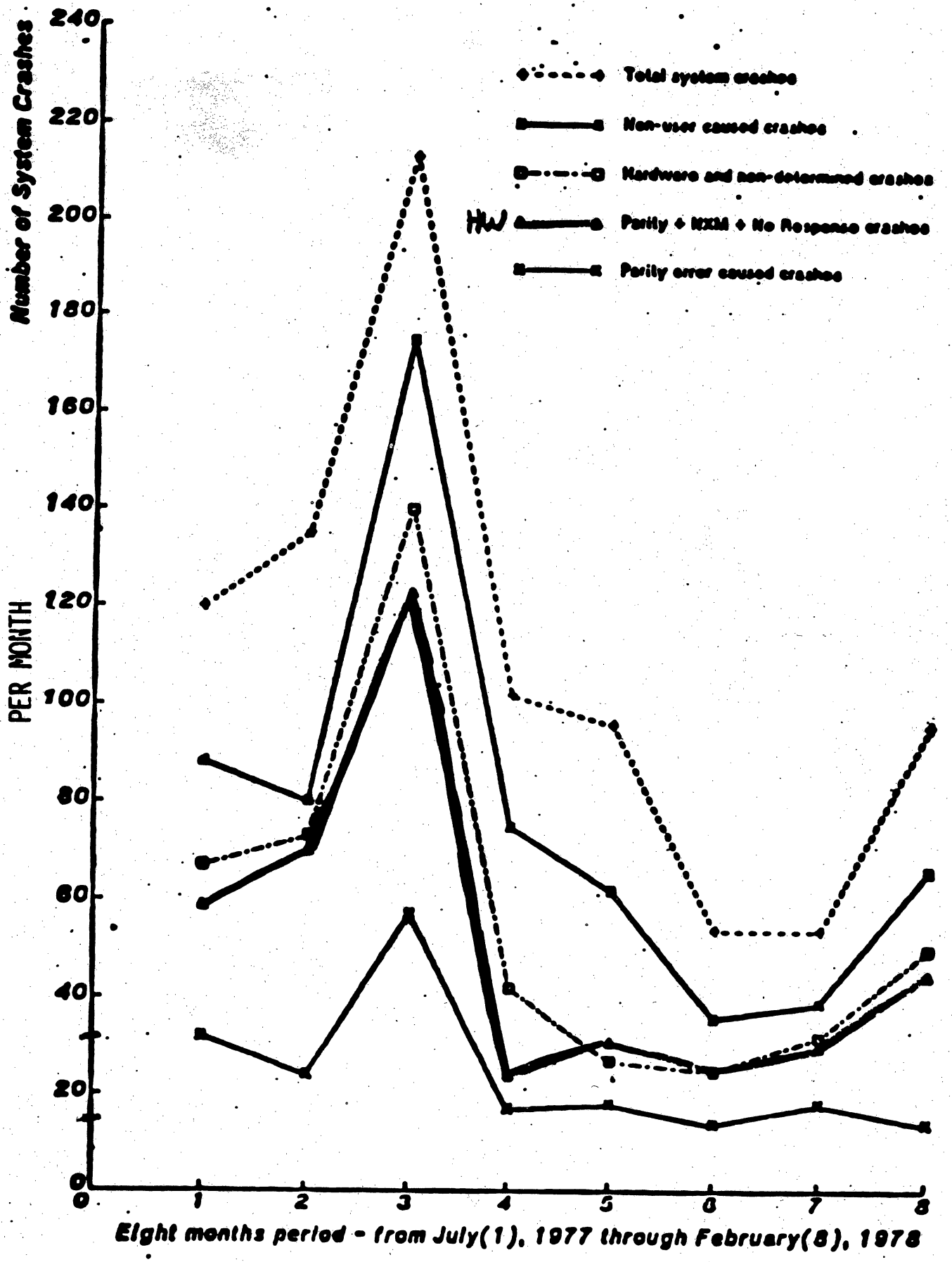
- Transients 20-50 times more frequent than hard failures
- Transient causes non-classic errors
  Transients have different extents
- Transients follow a decreasing failure rate
  ⟶ clustering
- Assuming exponential reliability function for transients when actual is Weibull can lead to incorrect conclusions.
- Software/Hardware ratio about 2:1
- Recovery mechanisms can prevent 80% of system crashes

Transient - due to temporary environment conditions.

- no physical damage

System Technology	Detection Mechanism	MTBE per Processor	MTTF per Processor	MTBE/MTTF
CMUA PDP-10,ECL	Parity	44 hrs.	800-1600 hrs.	0.03-0.06
Cm* LSI-11,NMOS	Diagnostics	128 hrs.	4200 hrs.	0.03
C.vmp TMR LSI-11	Crash	97-328 hrs.	4900 hrs.	0.02-0.07
Teletra, TTL	Mismatch	80-170 hrs.	1300 hrs.	0.08-0.13
1M x 37 RAM,MOS	(parity)	106 hrs.	1450 hrs.	0.07

Transients 20 times more prevalent than hard failures.


Number of System Crashes
PER MONTH
240
220
200
180
160
140
120
100
80
60
40
20
0
0
1
2
3
4
5
6
7
8
Total system crashes
Non-user caused crashes
Hardware and non-determined crashes
Parity + NXM + No Response crashes
Parity error caused crashes
HW
Eight months period - from July(1), 1977 through February(8), 1978

C.mmp

PROBLEMS OBSERVED (Hardware)

- Parity errors most dominant failure mode
- False "NXM's" (Non-Existing Memory)
- No response
- Clock nonmontonic
- Register specified not always selected
- Stack operations may push/pop too many (too few) operands on stack (JSR, RTS, RTI)
- Loss of interrupts, change interrupt levels
- Deadlocked memory ports due to incomplete bus cycle
- DMA overrun due to memory contention
- I/O devices do not return to an initial state when issued an illegal command

	Diagnostic test used					
Mode	Memory	Instr.	Trap	Slocal	Total	%Total
Burst	5	1	1	13	20	31.2
Simult	5	1	0	5	11	17.2
Isolat.	13	3	2	15	33	51.6
Total	23	5	3	34	64	100.0
%Total	35.9	7.8	4.7	51.6		

(Data collected from September, 1977 through April 1978)
(MTBE - 183.7 Mod-Hrs, @ 11, 758.4 total Mod-Hrs)

Table 4. Distribution of Transient Error on Cm* by mode of occurrence

	No. Ins.	No. Det.	%	Avg. Lat. (in μs)	Std. Dev. (in μs)	Tolerated w/perm. changes
Fault Duration						
1 cycle	493	202	41	13,000	85,000	43
2 cycles	484	298	61	12,000	70,000	56
4 cycles	460	313	68	14,000	91,000	52
Bus Type						
Inst	1053	724	69	4100	42,000	122
Data	384	89	23	82,000	200,000	29
Line Faulted						
D0	131	79	60	4300	23,000	19
D7	131	78	60	1200	9000	9
D8	131	63	48	1600	10,000	11
D15	130	81	62	210	550	17
A1	131	72	55	450	960	16
A4	131	67	51	24,000	110,000	15
A8	129	76	59	1600	10,000	13
A12	131	94	72	81,000	210,000	58
LDS	131	97	73	98	520	8
UDS	130	92	71	51	200	8
DTACK	131	14	11	43,000	100,000	5
Total/Avg.	1437	831	58	13,000	82,000	151

	No. Ins.	No. Det.	%	Avg. Lat. (in μs)	Std. Dev. (in μs)	Tolerated w/perm. changes
Fault Duration						
1 cycle	517	343	66	5400	61,000	17
2 cycles	462	402	87	5200	60,000	27
4 cycles	475	442	93	1400	27,000	35
Bus Type						
Inst	1124	1097	98	38	36	62
Data	330	90	27	51,000	180,000	17
Line Faulted						
D0	132	110	83	74	210	4
D7	132	111	84	79	230	4
D8	132	106	80	91	480	5
D15	132	108	82	65	190	4
A1	132	112	85	110	410	9
A4	132	103	78	65	200	5
A8	132	104	79	160	1000	4
A12	132	115	87	39,000	160,000	39
LDS	132	108	82	41	190	3
UDS	134	105	78	50	200	2
DTACK	132	105	80	79	190	0
Total/Avg.	1454	1187	82	3800	51,000	79

Effect of Fault	Number of faults
Detected by MC68000	680
Detected by MC68000 (tolerated w/perm. changes)	117
No voted data error	133
No voted data error (tolerated w/perm. changes)	4
Voted data error	447
Voted data error (tolerated w/perm. changes)	30
Unrecoverable errors (unmaskable errors in data or inability to completely execute benchmarks)	26
Total	1437

Exp. 1 Det. By	Exp. 2 Det. By	No. Det.	Exp. 1 Lat.	Exp. 2 Lat.
MC68000	MC68000 (same latency)	103	19μs	19μs
MC68000	MC68000 (lesser latency)	117	7.8ms	5.0ms
MC68000	MC68000 (greater latency)	92	170μs	210μs
MC68000	SIS	122	23.0ms	9.5ms
MC68000	NONE	7	51.0ms	-
NONE	MC68000	38	-	29μs
NONE	SIS	159	-	63μs
NONE	NONE	152	-	-

WEIBULL FUNCTION

$$R(T) = e^{-(\lambda T)^{\alpha}}$$

$\lambda \equiv$ SCALE PARAMETER

$\alpha \equiv$ SHAPE PARAMETER

FOR $\alpha = 1$

$$R(T) = e^{-\lambda T} \quad \text{(EXPONENTIAL)}$$

$$\text{CDF} = F(T) = 1 - e^{-(\lambda T)^{\alpha}}$$

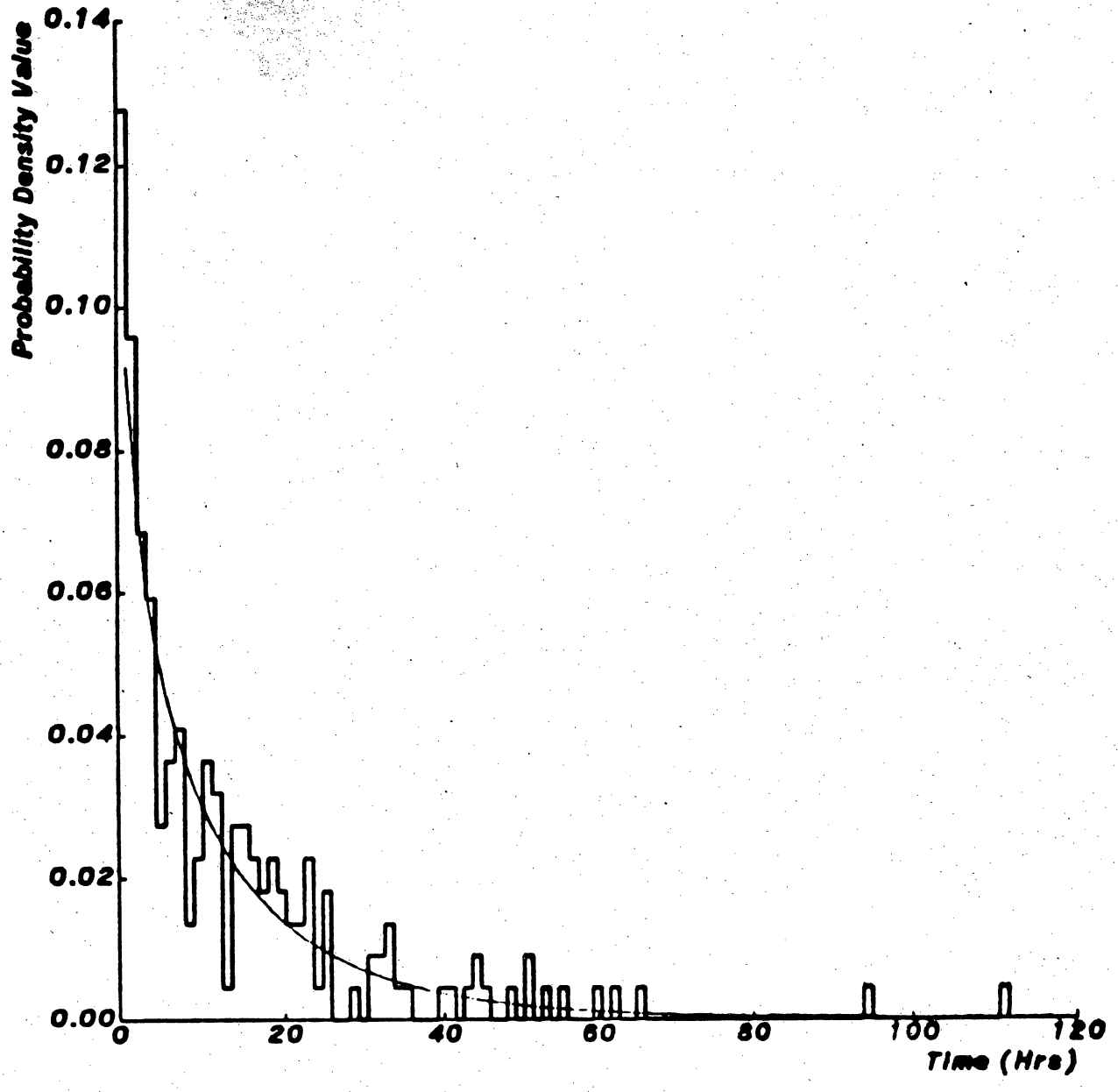


Figure 3-1: Distribution of TOPSC System Reloads

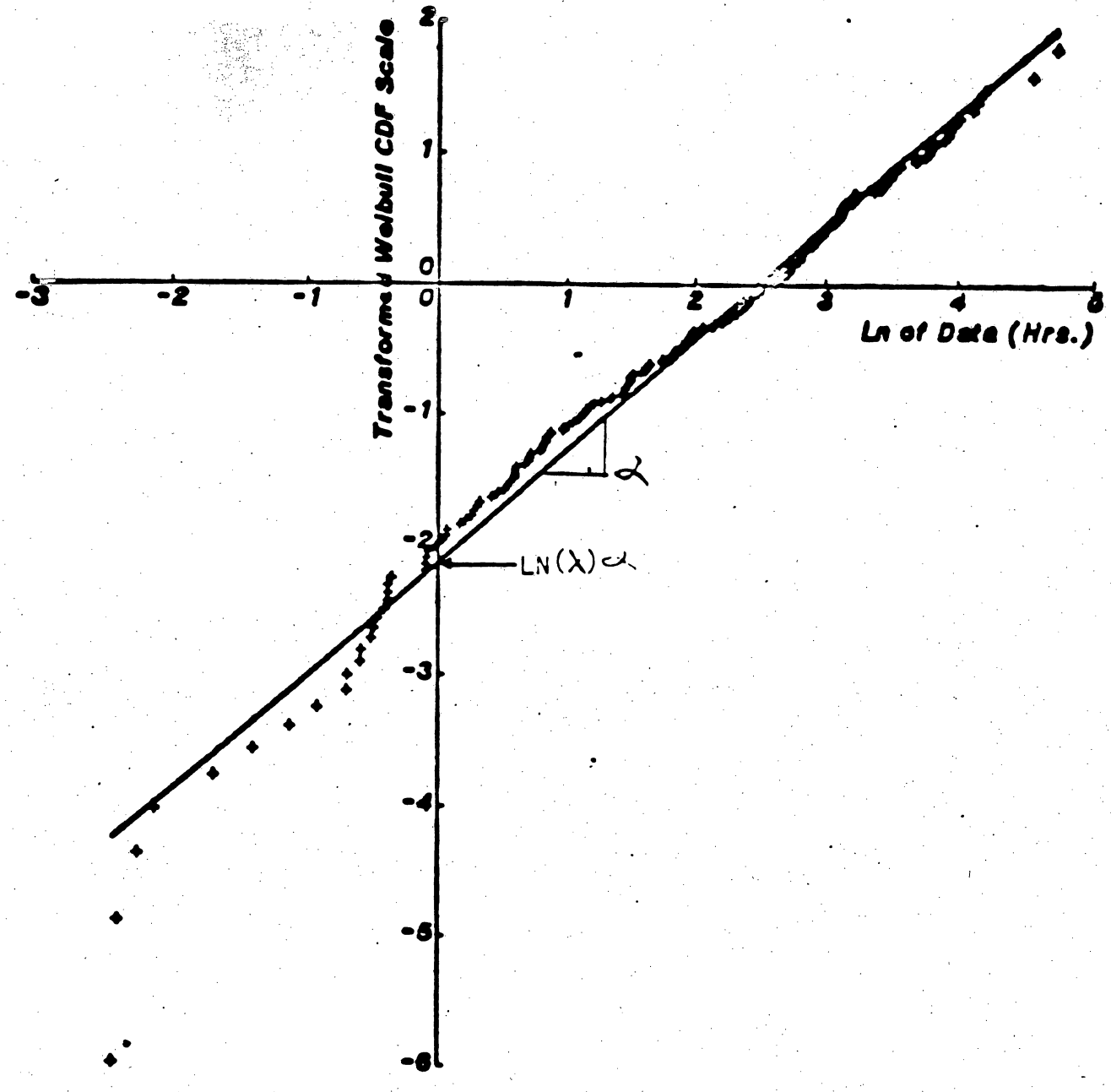


Figure 3-2: Weibull Plot of TOPSC System Reboots

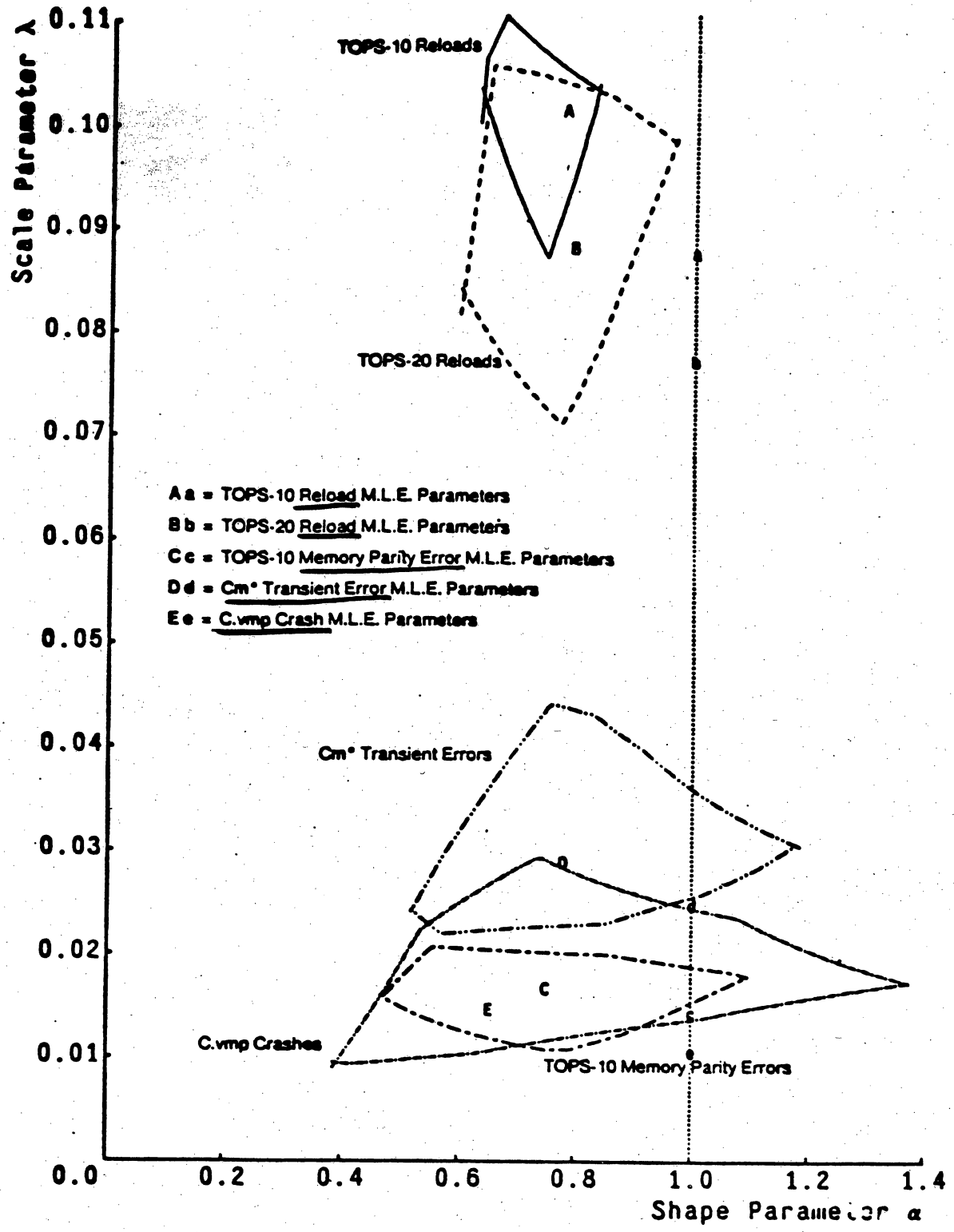


Figure 2-1: 90% Consonance Sets for Weibull Parameters

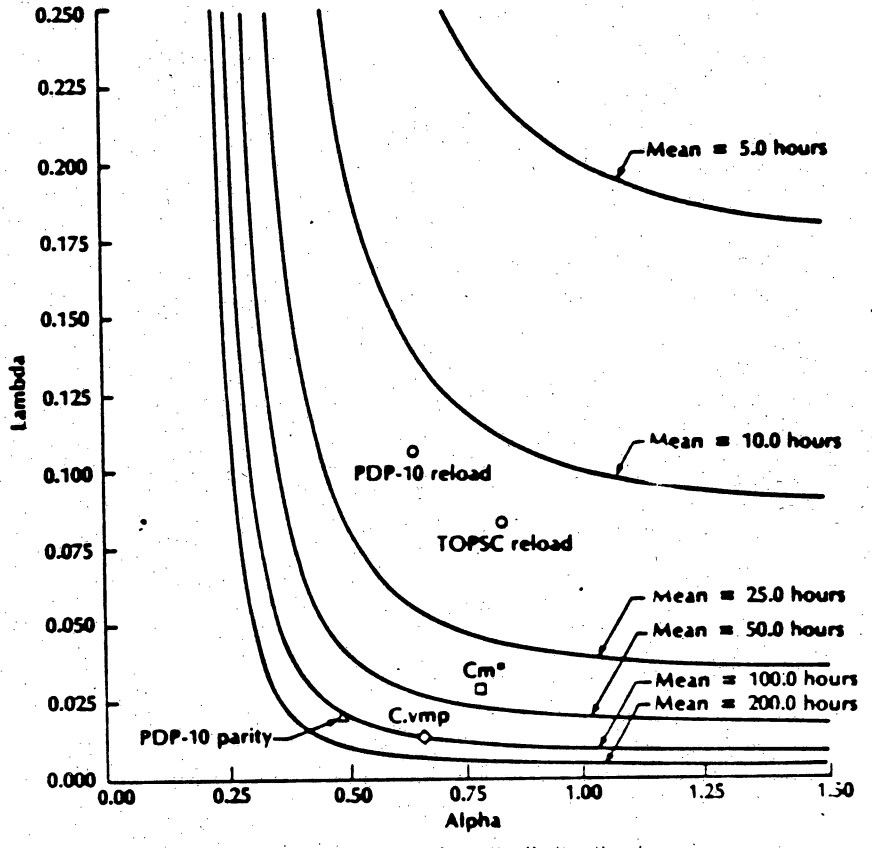


Figure 2–4. Means of Weibull distributions.

IBM 3081 vs KL-10

$\alpha = 0.647$ $\alpha = 0.639$

MTTC = 69.2 hrs MTTC = 13.5 hours

"in only 14% of the errors does the operating system stop processing."

λ_F = Mil 217

$\lambda_E = 20 \lambda_F$

$\lambda_{system} = 3 \lambda_E = 60 \lambda_F$

$\lambda_S = (2 K_{SR} + 1 K_{HR})$

K_{HR} → % unsuccessful retry

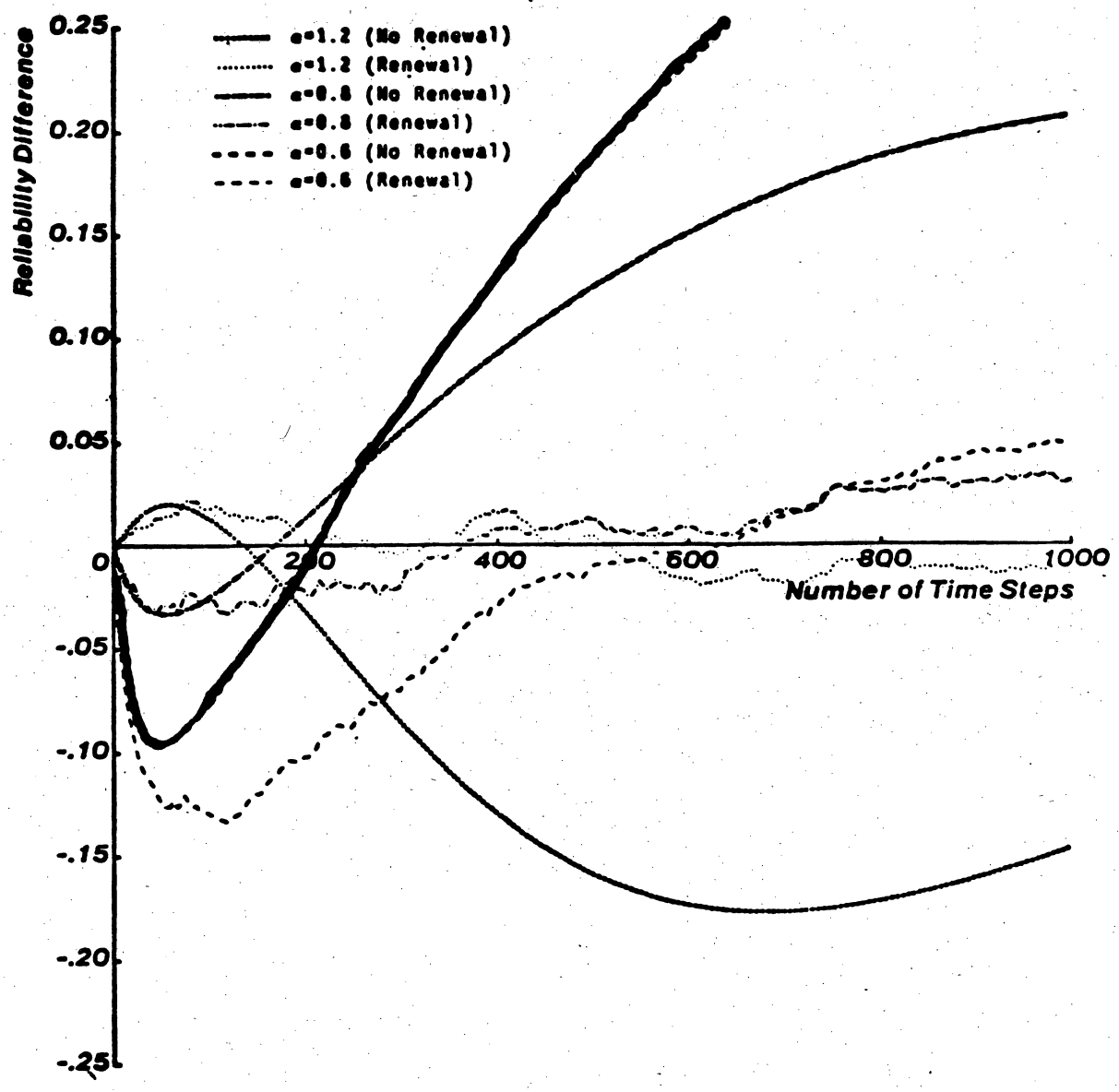
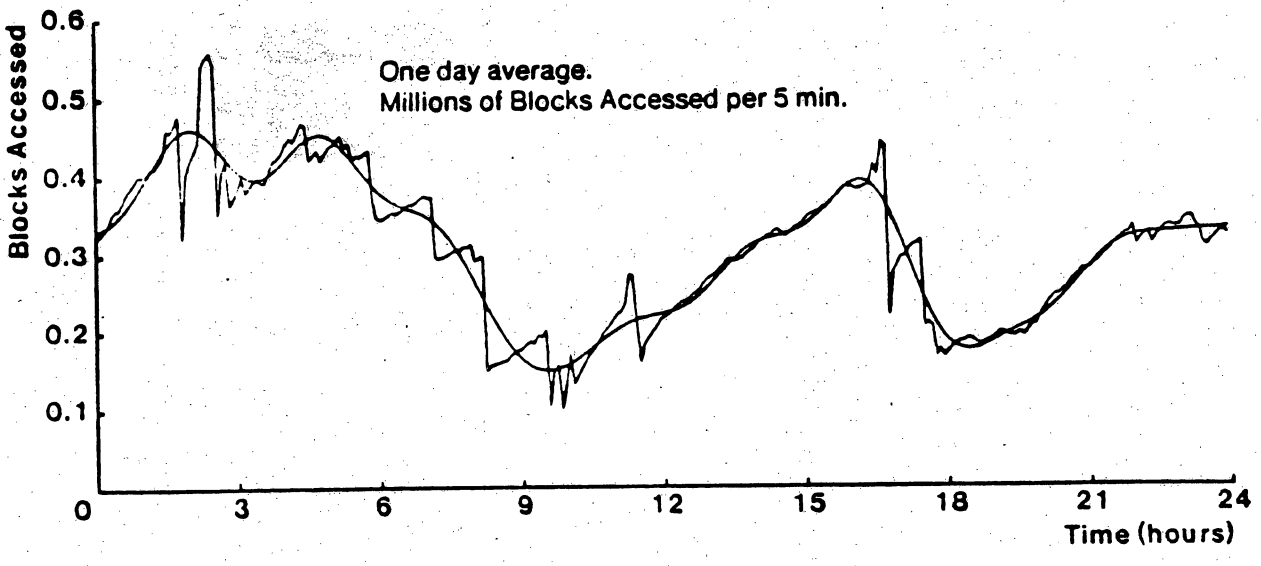
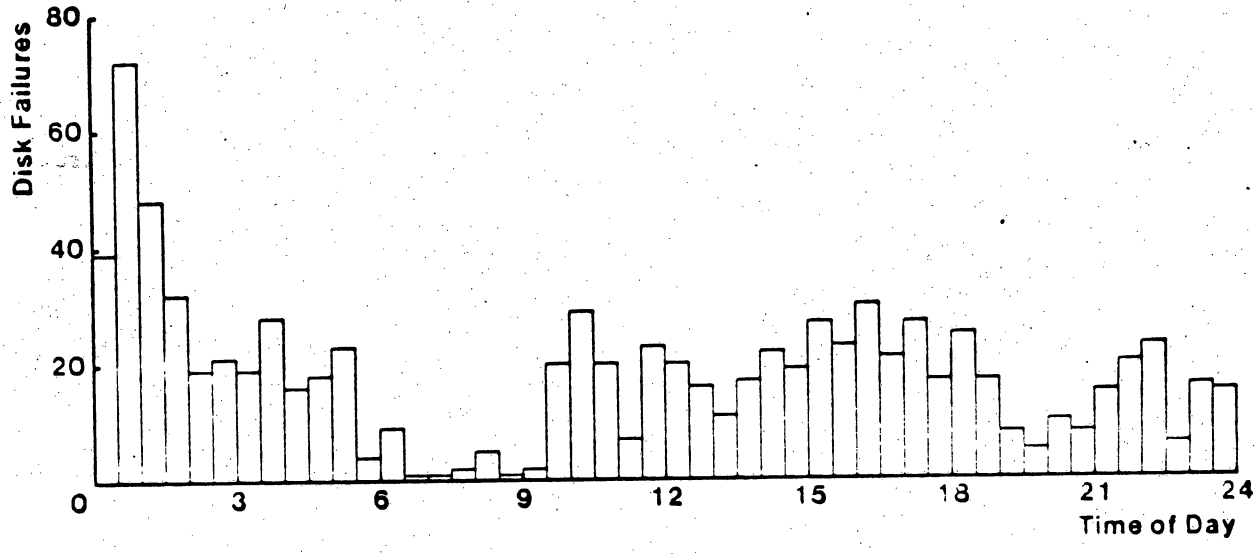


Figure 5-6: Reliability Difference of Duplex System Relative to $\alpha_e = 1.0$

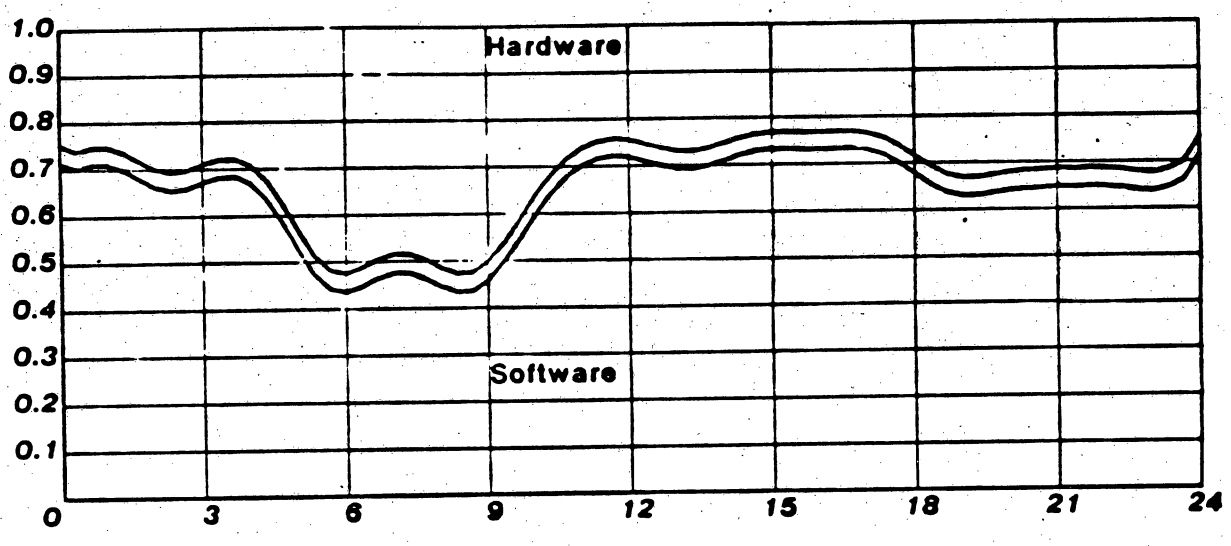

Blocks Accessed
0.6
0.5
0.4
0.3
0.2
0.1
0
One day average.
Millions of Blocks Accessed per 5 min.
3
6
9
12
15
18
21
24
Time (hours)


Disk Failures
80
60
40
20
0
3
6
9
12
15
18
21
24
Time of Day

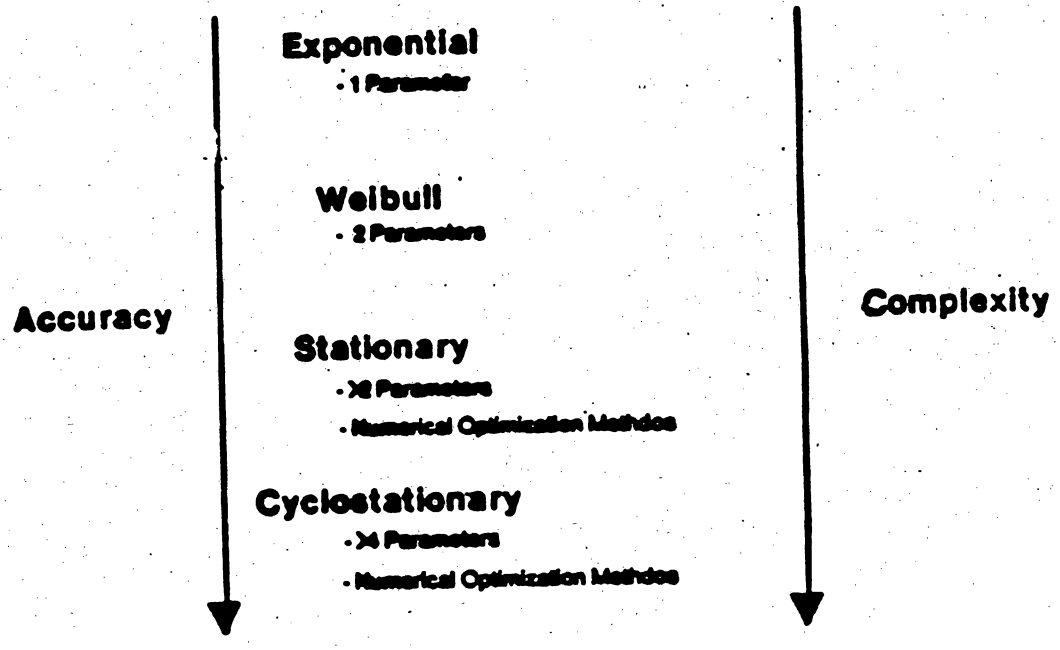
Applications : Software Reliability

- Given that a failure occurs at time t

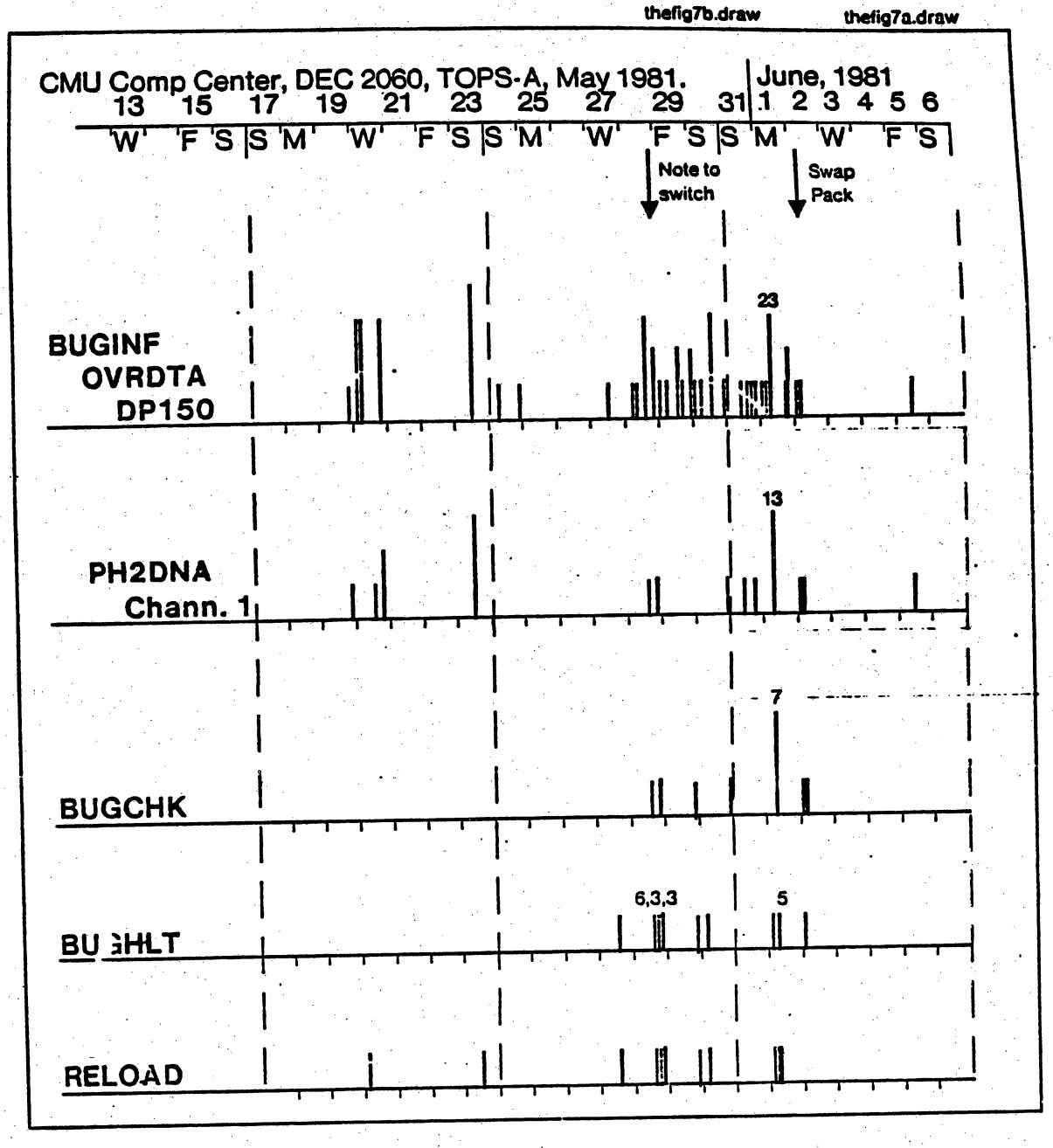
$$p_{sw}(t) = E\{ \frac{\lambda^{sw}(t)}{\lambda^{sw}(t) + \lambda^{hw}(t)} \}$$

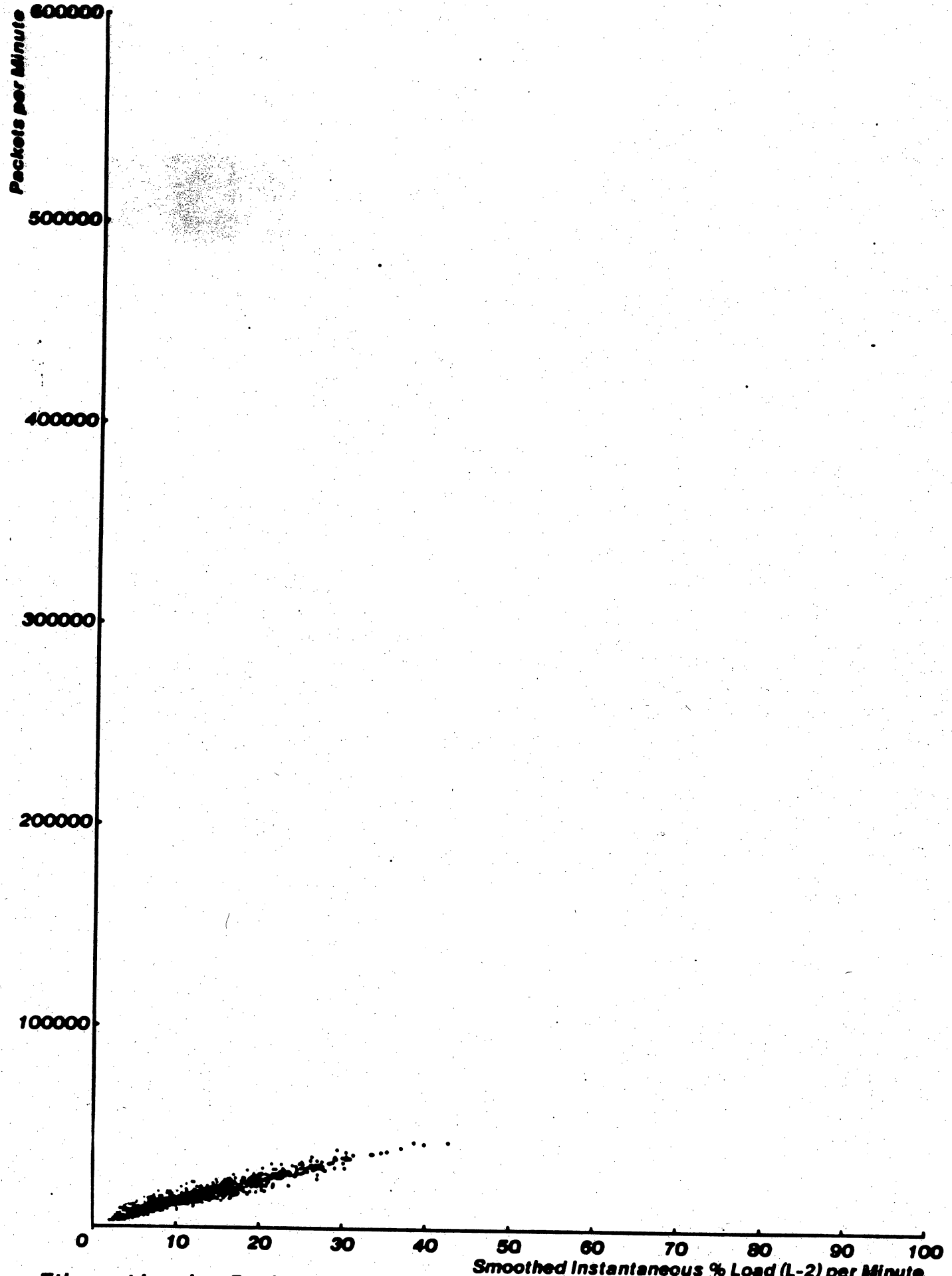


Model Complexity and Accuracy

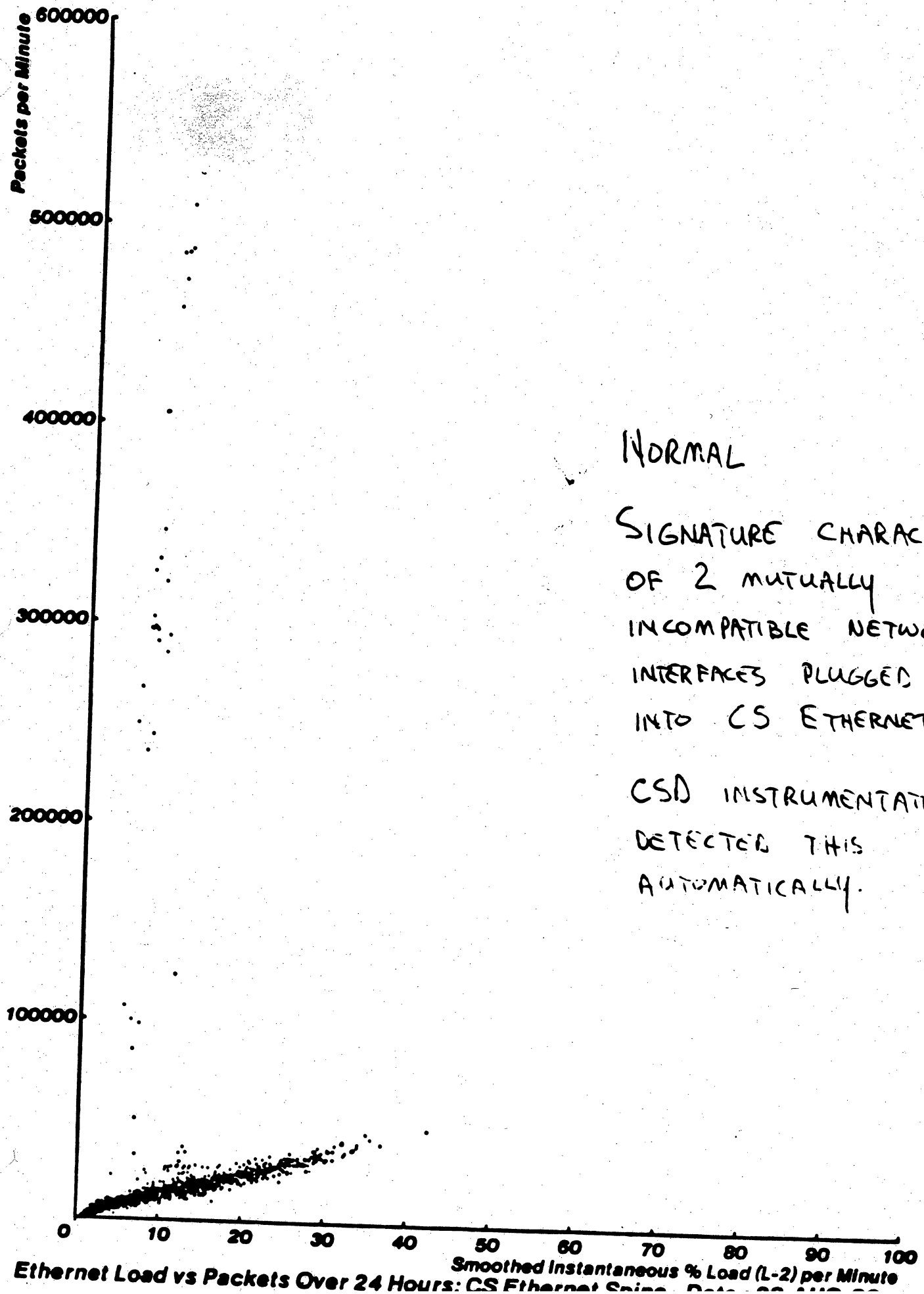


Manually Derived Trend Analysis
- RP06 on TOPS-A.





Ethernet Load vs Packets Over 24 Hours: CS Ethernet Spine - Date : 2-OCT-86



Ethernet Load vs Packets Over 24 Hours: CS Ethernet Spine

NORMAL

SIGNATURE CHARACTERISTIC OF 2 MUTUALLY INCOMPATIBLE NETWORK INTERFACES PLUGGED INTO CS ETHERNET.

CSD INSTRUMENTATION DETECTED THIS AUTOMATICALLY.

3.2) Hardware Fault Prediction

Component Failure Rates

Component failure distributions are usually assumed to have some sort of "failure rate"

- Constant Failure Rate (CFR)
- Decreasing Failure Rate (DFR)
- Increasing Failure Rate (IFR)
- Nonmonotic Failure Rate?

A time dependent failure rate is called a "hazard function".

$$Z(t)$$

Poisson Distribution: Conclusion

have obtained expressions for

$$\Pr\{k \text{ failures in time } [0,t]\}$$

defined reliability R(t) as

$$\Pr\{0 \text{ failures in time } [0,t]\}$$

Exponential: $Z(t) = \lambda$

$$R(t;\lambda) = e^{-\lambda t}$$

Weibull: $Z(t) = \alpha\lambda(\lambda t)^{\alpha-1}$

$$R(t;\alpha,\lambda) = e^{-(\lambda t)^{\alpha}}$$

General: Z(t)

$$R(t;Z(t)) = e^{-\int_0^t z(\tau)d\tau}$$

Nonredundant System Models

$$R_{sys}(t) = \prod_{i=1}^{n} R_i(t)$$

(assuming independent modules/components)

Exponential: $R_i(t) = e^{-\lambda_i t}$

$$R_{sys}(t) = e^{-\left(\sum_{i=1}^{n} \lambda_i\right) t}$$

Weibull: $R_i(t) = e^{-(\lambda_i t)^{\alpha_i}}$

$$R_{sys}(t) = e^{-\left(\sum_{i=1}^{n} (\lambda_i t)^{\alpha_i}\right)}$$

General: $R_i(t) = e^{-\int_0^t z_i(\tau)\,d\tau}$

$$R_{sys}(t) = e^{-\left(\sum_{i=1}^{n} \left(\int_0^t z_i(\tau)\,d\tau\right)\right)}$$

Parts Count Reliability Model

$$\text{Reliability}(t) = e^{-\lambda t}$$

λ = component failure rate

For a <u>nonredundant</u> system composed of n components:

$$\lambda_{system} = \sum_{i=1}^{n} \lambda_i$$

$$\text{Mean Time To Failure} = \frac{1}{\lambda_{system}}$$

For a general system:

$$MTTF = \int_0^\infty R(t)\, dt$$

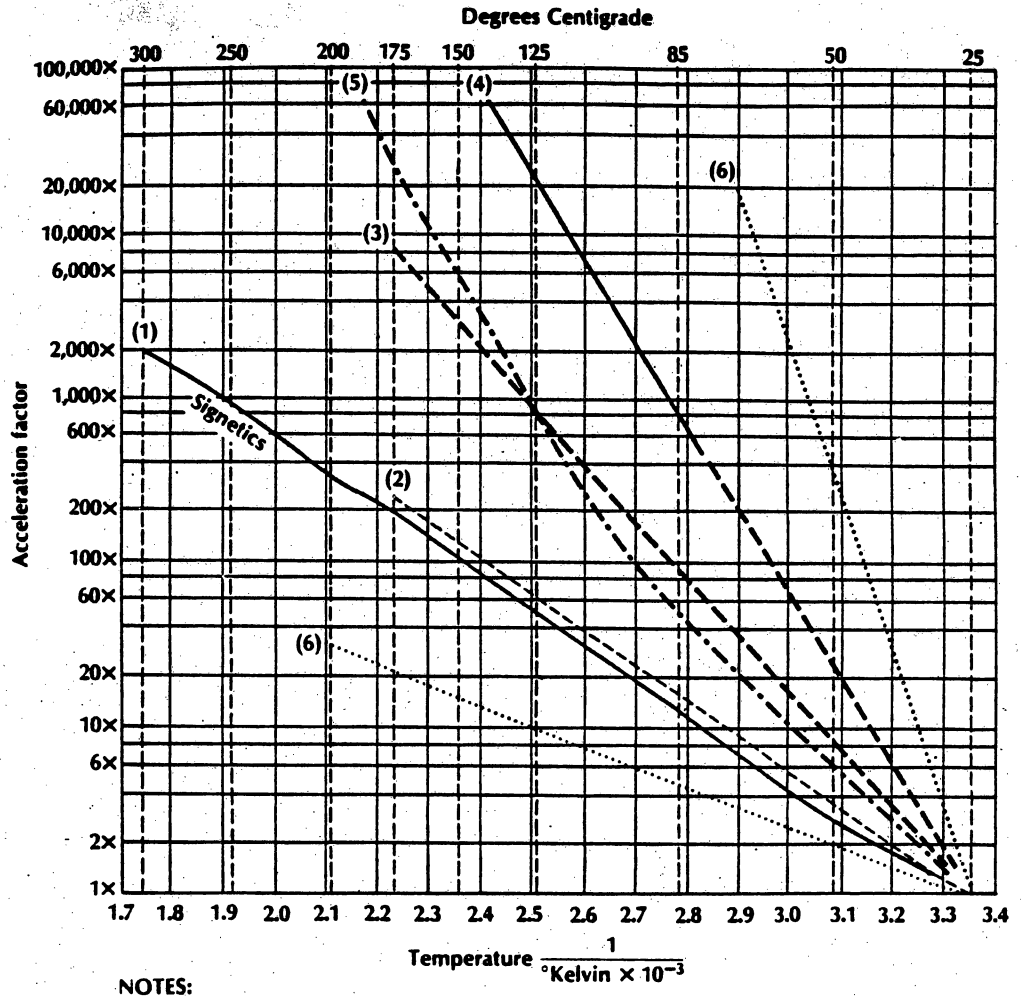
Reliability Modeling Calibration Problems

Components

- Not enough observed failures
  Need 10^7 to 10^8 device hours
- Temperature acceleration testing

System

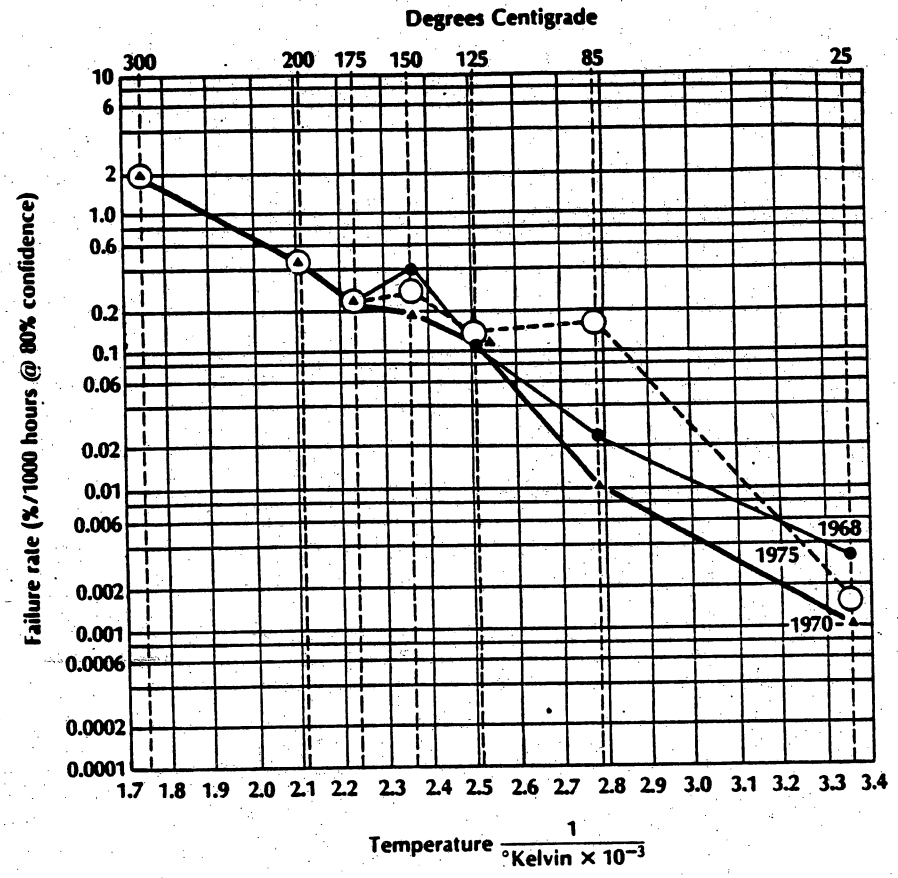
- Environmental factors of individual systems
- Process maturity



NOTES:

1. Calculated from the Signetics Failure Rate vs. Temperature Graph in [Signetics, 1975]. Signetics uses acceleration factors of 15 (for +85°C), 100 (for +150°C), 200 (for 175°C), 350 (for +200°C), 970 (for +250°C) and 2100 (for +300°C) to relate to +25°C equivalent ambient temperature. The +25°C to +125°C segment of the graph is based primarily on operating life data. The segment of the graph above +125°C is based on high temperature storage data. The graph equates to an "activation energy" E_a = 0.41eV.
2. Calculated from MIL-HDBK-217B, 20 September, 1974. Table 2.1.5.4. for πT_j vs T_j values. The graph equates to an "activation energy" E_a = 0.41eV and is applicable to all bipolar digital (except ECL) in the normal mode of operation.
3. Calculated from MIL-HDBK-217B, 20 September, 1974. Table 2.1.5.4. for πT_j vs. T_j values. The graph equates to an "activation energy" E_a = 0.70eV and is applicable to all MOS, all linear, and bipolar ECL devices in the normal modes of operation.
4. Calculated from MIL-STD-883A, 15 November 1974. Figures 1005-4 and 1015-1 by extrapolating the time temperature regression graph from +78°C back to +25°C. The MIL-STD-883A graph is the Bell Telephone Laboratories graph (Specification A-8-689143, 16 January 1974, etc) and as such applies to storage and operating T_j values and primarily surface inversion failure mechanisms. The graph equates to an "activation energy" E_a = 1.02eV.
5. This curved graph is the result of plotting the "rule of thumb" that failure rates (hence acceleration factors) double for every + Δ 10°C.
6. All competitor data (available to Signetics) produced graphs falling within these two boundaries. The two boundaries equate to "activation energies" E_a = 0.23eV (for lower graph) and E_a = 1.92eV (for the top graph).

Figure 2–5. Failure rate acceleration factor vs. temperature graphs: Signetics and others.



		1968		1970		1975	
		Cat.	Both (●)	Cat.	Both (▲)	Cat.	Both (◯)
HTSL	300°C	2.1	2.1	2.1	2.1	2.1	2.1
HTSL	200°C	0.22	0.48	0.20	0.46	0.2	0.46
HTSL	175°C	0.21	0.29	0.20	0.28	0.2	0.28
HTSL	150°C	0.32	0.40	0.182	0.20	0.155	0.24
HTSL & HTOL	125°C	0.088	0.117	0.101	0.125	0.110	0.147
HTOL	85°C	0.022	0.022	0.013	0.013	0.104	0.160
HTOL	25°C	0.0029	0.0038	0.0011	0.0013	0.0013	0.0017

NOTE:

1. The graphs were constructed to aid in the analysis and dramatization of the effect of the constituent parts of the failure rate equation. All tabulated failure rates were obtained from [Signetics, 1975] by combining life test data for like temperatures. Note that life test results of various die process technologies were indiscriminately summed together for this study.

Figure 2–6. Assessed failure rate vs. temperature graphs from 1963 to 1975 for catastrophic plus degradational failures.

Mil-HDBK-217

$$\lambda = \pi_L \pi_Q [C_1 \pi_T \pi_V \pi_{PT} + (C_2 + C_3) \pi_E]$$

π_L	Learning Factor	1, 10
π_Q	Quality Factor	0.5, 35
π_T	Temperature Acceleration	0.1, 50
π_V	Voltage Stress	1, 3
π_{PT}	PROM Programming Technique	1, 7
π_E	Environmental Factor	1, 10
C_1, C_2	Device Complexity	0.001, 0.2
C_3	Package Complexity	0.02, 0.2

TQ

Level		TQ
S	Mil M-38510, S	0.5
C	Mil M-38510, C	8.0
C-1	Mil STD-883	13.0
D	Commercial, hermetically sealed	17.5
D-1	Commercial	35.0

π_T

$$T_j = T_c + \theta_{jc} P$$

T_c = case temperature

θ_{jc} = case thermal resistance

P = worst-case power dissipation

Default θ_{jc}

Package Type	Number Pins	θ_{jc}
Hermetic DIP	≤ 22	30
	> 22	25
Nonhermetic DIP	≤ 22	125
	> 22	100

$$\pi_T = 0.1\, e^{\left(-A\left[\frac{1}{T_j+273} - \frac{1}{298}\right]\right)}$$

A

TTL, ECL	Hermetic	4635
	Nonhermetic	5214
Low Power Schottky TTL	Hermetic	5214
	Nonhermetic	5794
NMOS	Hermetic	6373
	Nonhermetic	9270

Example Values of π_T

T_j	TTL		LSTTL		NMOS	
	H	NH	H	NH	H	NH
40°C	0.21	0.26	0.26	0.28	0.28	0.44
70°C	0.77	1.0	1.3	1.7	1.7	5.9
90°C	1.6	2.3	3.25	4.6	4.6	26.2

π_V

For CMOS with supply voltage 12 - 15.5 V

$$\pi_V = 0.11\, e^{0.168\, V_s\, (T_j + 273)/298}$$

otherwise 1.0

π_{PT}

For bipolar PROM

$$\pi_{PT} = 0.985 + 9.5 \times 10^{-5}(B)$$

π_E

Ground benign = Computer Room	1.0
Ground Fixed = Factory Floor	2.5
Ground mobile = Jeep	4.0
Missile Launch	10.0

Table E–1. Additional environmental classes in MIL-HDBK-217C.

Class	*Example*	*Value*
Airborne, inhabited, transport	Cockpit of long-mission aircraft	2.8
Airborne, inhabited, fighter	Cockpit of high-performance aircraft	5.6
Airborne, uninhabited, transport	Equipment bay of long-mission aircraft	4.2
Airborne, uninhabited, fighter	Equipment bay of high-performance aircraft	8.4

ROM

$$C_1 = .285\ (10)^{-3}(B)^{0.603}$$
$$C_2 = .08\ (10)^{-3}(B)^{0.646}$$

RAM

$$C_1 = .995\ (10)^{-3}(B)^{0.603}$$
$$C_2 = .28\ (10)^{-3}(B)^{0.644}$$

for $B \leq 16{,}384$.

In the case of LSI devices, the coefficient of N_G in the exponent decreased by a factor of four from MIL-HDBK-217B. C_1 and C_2 decreased by a factor of four for ROMs and a factor of two for RAMs. These decreases reflect the increased reliability of semiconductor components as fabrication experience grows.

217C NOTICE 1 MODEL

The failure rate, λ, in failures per million hours for monolithic MOS and bipolar chips takes the form of:

$$\lambda = \pi_L \pi_Q [C_1 \pi_T \pi_V \pi_{PT} + (C_2 + C_3)\pi_E]$$

Values for each factor will be discussed in turn.

π_L

The learning factor has a value of 10 if the device is new, if there are major changes in the fabrication process, or if the fabrication process is being restarted after an extended interruption. Otherwise the value of π_L is 1.0.

π_Q

The quality factor is a function of the amount of device screening. Table E-2 lists the values of π_Q. Note the decreases in π_Q (ranging from a factor of 1.5 to 8.5 lower) over the values, for the same classes, in MIL-HDBK-217B.

π_T

The temperature acceleration factor is a function of device technology, package type, case temper-

Table E–2. Quality factors for MIL 217C notice 1.

Quality Level	*Screening Standard*	π_Q
S	MIL-M-38510, Class S	0.5
B	MIL-M-38510, Class B	1.0
B-1	MIL-STD-883, Method 5004, Class B	3
B-2	Same as B-1 with some tests waived	6.5
C	MIL-M-38510, Class C	8.0
C-1	MIL-STD-883, Method 5004, Class C	13.0
D	Commercial, hermetically sealed	17.5
D-1	Commercial	35.0

ature, and power dissipation.

$$\pi_T = 0.1e^{[-A[(1/(T_j+273))-(1/298)]]}$$

where A is a function of technology and package type as given in Table E-3 and T_j is the worst-case junction temperature. T_j is given by:

$$T_j = T_C + \Theta_{JC} P$$

where:

- T_C is the case temperature (°C)
- Θ_{JC} is the junction to case thermal resistance (°C/watt) for a device soldered onto a printed circuit board.
- P is the worst-case power dissipation.

If Θ_{JC} is unknown, the values in Table E-4 may be used. Some useful values of π_T are given in Table E-5. Early experience suggests that 217C Notice 1 is very sensitive to the junction temperature calculation. In particular, the default values for Θ_{JC} lead to questionably large values for T_j, especially in the case of nonhermetic packages.

π_V

The voltage stress factor, π_V, is 1.0 for all technologies other than CMOS. π_V is also 1.0 for CMOS with $V_{DD} = 5$ volts. For supply voltage between 12 and 15.5 volts:

$$\pi_V = 0.11e^{x}$$

Table E–3. Technology and package parameter, A, used in calculation of π_T.

Technology	*Package Type*	*A*
TTL, ECL	Hermetic	4635
TTL, ECL	Nonhermetic	5214
Schottky TTL	Hermetic	5214
Schottky TTL	Nonhermetic	5794
Low Power Schottky TTL, PMOS	Hermetic	5794
Low Power Schottky TTL	Nonhermetic	6373
PMOS	Nonhermetic	8111
NMOS	Hermetic	6373
NMOS	Nonhermetic	9270
CMOS, Linear	Hermetic	7532
CMOS, Linear	Nonhermetic	10429

Table E–4. Default values for thermal resistance, θ_{JC}.

Package Type	*Number of Pins*	θ_{JC}
Hermetic DIP	≤ 22	30
Hermetic DIP	> 22	25
Nonhermetic DIP	≤ 22	125
Nonhermetic DIP	> 22	100

Table E–5. Some useful values of π_T.

Junction	*Low-Power TTL*		*NMOS*		*TTL*	
Temperature	*Hermetic*	*Nonhermetic*	*Hermetic*	*Nonhermetic*	*Hermetic*	*Nonhermetic*
25 (°C)	0.1	0.1	0.1	0.1	0.1	0.1
40 (°C)	0.26	0.28	0.28	0.44	0.21	0.26
70 (°C)	1.3	1.7	1.7	5.9	0.77	1.0
90 (°C)	3.25	4.6	4.6	26.2	1.6	2.3

where:

$$x = \frac{0.168 V_s (T_j + 273)}{298}$$

with V_s the supply voltage.

π_{PT}

π_{PT} is the PROM programming technique factor. π_{PT} is 1.0 for all devices except PROMs. For bipolar PROMs:

$$\pi_{PT} = 0.985 + 9.5 \times 10^{-5}(B)$$

where B is the number of bits. For MOS PROMs:

$$\pi_{PT} = 0.95 + 7.5 \times 10^{-5}(B)$$

π_E

The application environment factor depends on the operational environment as indicated in Table E-6.

Table E–6. Environmental factor, π_E.

Environment	*Example*	π_E
Ground, benign	Computer room	1.0
Space flight	Satellite	1.0
Ground, fixed	Factory floor	2.5
Airborne, inhabited, transport	Cockpit of long-mission aircraft	3.5
Airborne, inhabited, fighter	Cockpit of high-performance aircraft	7.0
Naval, sheltered	Bridge of a surface ship	4.0
Ground, mobile	Jeep	4.0
Airborne, uninhabited, transport	Equipment bay of long-mission aircraft	4.0
Airborne, uninhabited, fighter	Equipment bay of high-performance aircraft	8.0
Naval, unsheltered	Engine room of a surface ship	5.0
Missile, launch	Missile	10.0

Table E–7. MIL-HDBK-217C, notice 1 complexity factors.

Density/Function	C_1	C_2
Bipolar SSI/MSI	$7.48\ (10)^{-4}(N_G)^{0.654}$	$2.19\ (10)^{-4} \cdot (N_G)^{0.364}$
MOS SSI/MSI	$2.17\ (10)^{-3}(N_G)^{0.357}$	$3.11\ (10)^{-4} \cdot (N_G)^{0.178}$
Linear	$1.57\ (10)^{-3}(N_T)^{0.780}$	$8(10)^{-4}(N_T)^{0.535}$
Bipolar LSI ($100 < N_G < 20{,}000$)	$1.48\ (10)^{-3}(N_G)^{0.506}$	$3.2\ (10)^{-4}(N_G)^{0.279}$
MOS LSI ($100 < N_G < 20{,}000$)	$1.75\ (10)^{-3}(N_G)^{0.4}$	$2.52\ (10)^{-4} \cdot (N_G)^{0.226}$
Bipolar RAM ($B \leq 16{,}384$)	$2.2\ (10)^{-3}(B)^{0.576}$	$4(10)^{-5}(B)^{0.554}$
Dynamic MOS RAM ($B \leq 65{,}536$)	$5(10)^{-4}(B)^{0.610}$	$3(10)^{-5}(B)^{0.585}$

(*Table continues on next page*)

Table E–7 —*Continued*

Density/Function	C_1	C_2
Static MOS RAM ($B \leq 65{,}536$)	$6(10)^{-4}(B)^{0.653}$	$4(10)^{-5}(B)^{0.609}$
Bipolar ROM, PROM ($B \leq 65{,}536$)	$8.8\ (10)^{-4}(B)^{0.388}$	$4.5\ (10)^{-5}(B)^{0.378}$
MOS ROM, PROM ($B \leq 65{,}536$)	$1.2\ (10)^{-3}(B)^{0.425}$	$6.6\ (10)^{-5}(B)^{0.399}$

Key: N_G is the number of gates
N_T is the number of transistors
B is the number of bits

C_1, C_2

C_1 and C_2 are a function of the device complexity and the device function as given in Table E-7.

C_3

C_3 is a function of package complexity as given in Table E-8.

Table E–8. MIL-HDBK-217C, notice 1 package complexity.

Package Type	C_3
Hermetic DIPs with solder or weld seals	$2.8\ (10)^{-4}(N_P)^{1.08}$
Hermetic DIPs with glass seals	$9(10)^{-5}(N_P)^{1.51}$
Nonhermetic DIPs	$2(10)^{-4}(N_P)^{1.23}$
Hermetic Flatpacks	$3(10)^{-5}(N_P)^{1.82}$
Hermetic Cans	$3(10)^{-5}(N_P)^{2.01}$

Note: N_P is the number of pins on the package connected to the device substrate.

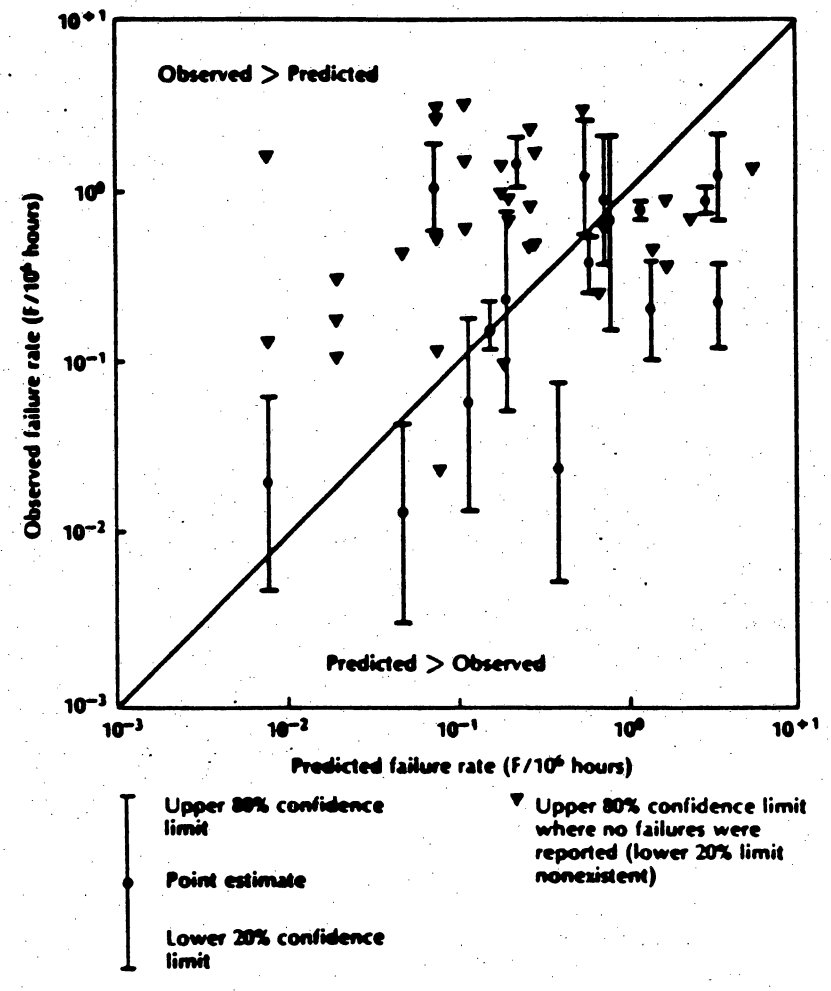


Figure 2–8. Digital-TTL integrated circuit observed vs. MIL-HDBK-217B predicted failure rates of SSI (1-10 gates) and MSI (11-25 gates) complexity.

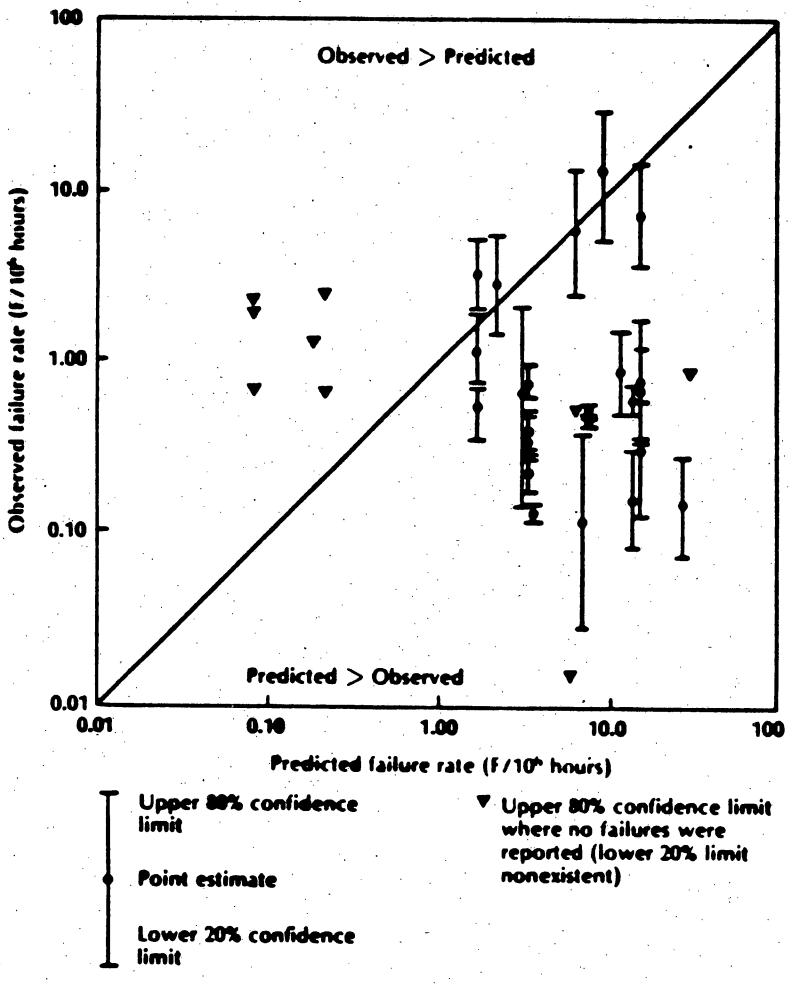


Figure 2–9. Random access memory (RAM) observed failure rates vs. MIL-HDBK-217C predicted failure rates.

Data Collection

$R(t) = e^{-\lambda t}$

Module	Complexity (Chips)	# of Modules	Total time (Hours)	Total failures	MTBF (Hours)
Kbus	138	3	36696	8	4587
Pmap	106	3	37416	12	3118
Mmicro	116	6	68328	4	17082
Mdata	142	3	37080	2	18540
Linc	116	3	22608	0	-
LSI-11	68	14	163200	10	16320
Slocal	126	10	120720	5	24144
4K memory	56	21	260568	5	52003.6
16K memory	104	10	122280	5	24456
Slu	28	17	223248	5	44649.6
Power board	6	16	195456	3	65152
Refresh	14	16	162912	0	

TABLE 2

COMPARISON OF MODIFIED MIL-HDBK-217B MODEL

WITH

EXPERIMENTAL DATA

MODULE	OBSERVED FAILURE RATE (PER 10^6 HOURS)	BEST FIT MODIFICATION TO MIL-217B	PREDICTED FAILURE RATE WITH MODIFIED MIL-217B
KBUS	218	Q = 150/16	413
PMAP	320.7	Q = 150/16	333.7
MMICRO	58.5	Q = 16/16	26.6
MDATA	53.9	Q = 16/16	35.4
LINC	-		
LSI-11	61.3	Q = 16/16	29.9
SLOCAL	41.4	Q = 16	31.8
4K MEMORY	19.2	Q = 16/16	23.1
16K MEMORY	40.9	Q = 16/16	74.1
SLU	22.4	Q = 150/16	43.9
POWER BOARD	15.3	Q = 150/16	9.1
REFRESH	-		

TABLE 6

Table 2–10. Data from transient errors in a time-sharing file system.

Category	O_i	E_i
0–5	548	429.20
5–10	148	219.15
10–15	63	111.89
15–20	35	57.13
20–25	28	29.17
25–30	18	14.89
30–35	12	7.60
35–40	6	3.88
40–45	3	1.98
45–50	1	1.01
50–55	3	0.5178
55–60	2	0.2639
60–65	1	0.1347
65–70	1	0.06881
70–75	1	0.03514
75–80	1	0.01794
80–85	1	0.009160
85–90	1	0.004690
90–95	1	0.002395
95–100	1	0.001215
100–105	1	0.000627

Table 2–11. Combining categories from Table 2–10.

Category	O_i	E_i	$(O_i - E_i)^2/E_i$
0–5	548	429.20	32.88
5–10	148	219.15	23.10
10–15	63	111.89	21.36
15–20	35	57.13	8.57
20–25	28	29.17	0.04
25–30	18	14.89	0.64
30–35	12	7.60	2.53
35–∞	25	7.93	36.74
		Total χ^2 =	125.86

Table 2–12. Time between crashes for a time-sharing system during one month of operation.

Category (hours)	O_i
0–1	6
1–2	3
2–3	5
3–4	2
4–5	7
5–6	5
6–7	1
7–8	1
8–9	3
9–10	4
10–11	2
11–12	1
12–14	2
14–15	2
15–16	1
16–17	1
17–18	3
18–21	1
21–24	4
24–29	1
29–38	3
38–75	2

Table 2–13. Combining categories from Table 2–12.

Category (hours)	O_i	E_i	$(O_i - E_i)^2/E_i$
0–2	9	9.97	0.09
2–4	7	8.17	0.16
4–6	12	6.79	3.97
6–8	2	5.67	2.37
8–11	9	6.80	0.70
11–15	5	6.66	0.41
15–20	5	5.61	0.06
20–28	6	5.14	0.14
28–∞	5	5.13	0.003
		Total $\chi^2 = 7.95$	

Lambda - A Design for Reliability Support Tool

1. *follow design's structure:*
   - systems descriptions top-down
   - analysis outputs display structure
2. *"split" components:*
   - function quantities of a part
3. *failure rates of internal elements of IC's:*
   - not directly, through CUST parts
   - will be added to Lambda in the future.
4. *analyze contribution by part type/number:*
   - SORTKEY, PARTSROLL
5. *accommodate component data hierarchy:*
   - calling parameters
   - part specifications
   - defaults
6. *support for all phases of design:*
   - default parameters
   - "generic" component types,
7. *supply models of standard reliability enhancement techniques:*
   - fault avoidance: MIL-217 models, library function for Θ_{ja}
   - fault detection: not now, can be added through library
   - fault tolerance: table of library functions shown previously
8. *ability to cooperate with other CAE tools:*
   - PRINT command
   - no general reading ability

Example: PDP-8 Redesign

```
module [pdp8_e:
        1, $cpu:
        1, $memext.timshr;
        1, $16k.memory;
 ];

module [cpu:
                1, $data_part;
                1, $cpu.busconn;
                1, $bus.lds.MB320;
                1, $control.logic;
        ];

module [data_part:
                1, $registers;
                1, $data.path;
        ];

module [registers:
                1,[A.MQ.MB.PC.MA:       ! on mb300
                        15,DEC8271(,,@ic_theta (pkg, air_flow))  ];
                1,[LINKBIT:             ! on mb310
                        1F,7474(,,@ic_theta (pkg, air_flow))  ]
        ];

module [data.path:
                1,[adder:   3,7483(,,@ic_theta (pkg, air_flow))  ];
                1,[t.c.1.0:   3,7487(,,@ic_theta (pkg, air_flow))  ];
                                        !true.compl.one.zero
                1,[path.shunt:  12F,74153(,,@ic_theta (pkg, air_flow));
                                29F,DEC8266(,,@ic_theta (pkg, air_flow));
                                12F,DEC8235(,,@ic_theta (pkg, air_flow));
                                12F,74151(,,@ic_theta (pkg, air_flow));
                                12F,7400(,,@ic_theta (pkg  air_flow))
                  ]
         ];

module [cpu.busconn:
                1,[MB300.oc:    24F,DEC8881(,,@ic_theta (pkg, air_flow)) ];
                1,[MB310.oc:    8F,DEC8881(,,@ic_theta (pkg, air_flow)) ];
          ];

module [bus.lds.MB320:    1,7400(,,@ic_theta (pkg, air_flow))  ];
```

PDP-8 redesign analysis
Nonredundant version failure rate analysis

Quantity) Module	Lambda - single Module	% Lambda - all module copies
1) PDP8_E	63.7661	100.000
1) CPU	9.7628	15.310
1) DATA_PART	4.6454	47.583
1) REGISTERS	1.8614	40.069
1) A.MQ.MB.PC.MA	1.8374	98.713
1) LINKBIT	0.0240	1.287
1) DATA.PATH	2.7840	59.931
1) ADDER	0.6751	24.250
1) T.C.1.0	0.1291	4.638
1) PATH.SHUNT	1.9798	71.112
1) CPU.BUSCONN	0.1794	1.838
1) MB300.OC	0.1345	75.000
1) MB310.OC	0.0448	25.000
1) BUS.LDS.MB320	0.0238	0.244
1) CONTROL.LOGIC	4.9142	50.336
1) MB300.CTL	0.1224	2.491
1) MB310.CTL	1.4423	29.350
1) TIM_GEN	3.3496	68.159
1) MEMEXT.TIMSHR	2.1555	3.380
1) 16K.MEMORY	51.8478	81.309
1) MEMORY.CHIPS	51.3166	98.976
1) CONTROL	0.4639	0.895
1) BUS.CONN.OC	0.0673	0.130

Flags after module name:
"*" - possible default parameter change
"R" - module has redundant submodules
"RF" - module is defined by REL.FN

**** MTTF: 15682.3 hrs. ****
Est. number parts: 285.1 Est. number pins: 4316
Est. Costs: Net Part Cost: 0.0000
Repair Cost Rate : 0.0000 per million Hrs.
Mean Material Repair Cost: 0.0000
with 34 zero-cost part numbers

Sortkey summary statistics :

Key	Components	Net Lambda	% System Lambda
IC:RAM	48.0	51.3166	80.48
IC:SSI	156.8	4.1225	6.47
IC:MSI	80.3	8.3269	13.06
IC:LSI	0.0	0.0000	0.00

PDP-8 redesign analysis
Nonredundant version failure rate analysis

Partname summary statistics :

Part Name	Components	Net Lambda	% System Lambda
MK4096	48.0	51.3166	80.48
DEC8271	20.0	2.4499	3.84
74194	10.0	2.0921	3.28
DEC8266	9.3	0.8510	1.33
7474	15.0	0.7186	1.13
74151	13.0	0.7135	1.12
7483	3.0	0.6761	1.06
DEC8881	27.0	0.6054	0.95
DEC8235	6.0	0.5520	0.87
7400	23.0	0.5474	0.86
7404	15.8	0.4728	0.74
MSIMISC	6.0	0.3293	0.52
74153	6.0	0.3067	0.48
7440	12.0	0.2646	0.41
7402	8.0	0.2000	0.31
DEC8251B	3.0	0.1769	0.28
7410	8.0	0.1663	0.26
DEC6380	7.0	0.1570	0.25
DEC384	6.0	0.1345	0.21
SSIMISC	6.0	0.1345	0.21
7420	6.0	0.1323	0.21
7487	3.0	0.1291	0.20
74123	1.0	0.1068	0.17
7405	4.0	0.1055	0.17
DEC6314	4.0	0.0714	0.11
7412	3.0	0.0624	0.10
7430	3.0	0.0619	0.10
7417	2.0	0.0528	0.08
7486	1.0	0.0511	0.08
DEC97401	2.0	0.0448	0.07
DEC7384	1.0	0.0224	0.04
DEC8640	1.0	0.0224	0.04
7411	1.0	0.0208	0.03
DEC5314	1.0	0.0179	0.03
74183	0.0	0.0000	0.00
74174	0.0	0.0000	0.00
ECC18	0.0	0.0000	0.00
74157	0.0	0.0000	0.00
74154	0.0	0.0000	0.00

Default and specific parameters for this analysis
appear on the following page

3.3) Software Fault Prediction

The words *Software Reliability Model* have been used to describe :

- *Time domain* models, which attempt to estimate the remaining number of bugs in a software system after "X" months of debugging.
- *Structural models* which provide ways to estimate what fraction of all possible input data combinations may lead to a failure in a software module.
- *Axiomatic models*, in which software behavior is predicated to have certain properties analogous to other physical systems.

But there are no general methods for software reliability assesment from easily measurable variables in a running system.

Software Evaluation Models

- Up to 90% of computing dollars spent annually on software.
- Estimating manpower.
- Determine when goal achieved.

Basic Assumption

- Software failure rate proportional to number of bugs or design errors present.

Basic Assumption ignores

- Relative (data dependent) frequency of use.
- Workload.

Categories of Software Reliability

Models

- Time Domain
- Data Domain
- Axiomatic
- Other

Time Domain Models

- Software reliability proportional to the number of bugs (implementation errors) present - analogous to hardware MTTF.
- Removal of implementation errors increases MTTF.
- Correlation of bug-removal history with the time evolution of the MTTF value may allow prediction of when a given MTTF value will be reached.

Disadvantages

- Bug correction may generate more bugs which is not modeled.
- Bugs are not the only source of software unreliability (i.e., design errors, characterization of intended environment, etc.).

Time Domain Models

Shooman

$$R(t) = e^{-CE(r,T)}$$

- R - probability of no software failure during time interval [0,t].
- E - number of errors present in the system.
- T - number of months of debugging.
- C,E - estimated from results of the exerciser and the number of errors corrected.

Jelinsky - Moranda

Further assumptions:

- Errors removed immediately upon discovery.
- Debugging time between error occurrences exponential distributed.

Then time to discover i^{th} error measured from time of discovery of i-1th error is:

$$P(t_i) = \lambda(i)e^{-\lambda(i)t_i}$$

where $\lambda(i) = f(N-i+1)$

N = number of original errors

Littlewood

- Program execution simulated with a continuous-time Markov model switching among smaller programs.
- Use testing history of software modules to estimate the probable number of errors in each.

Data Domain Models

Nelson

- Sets of inputs E

$$E = \{E_i, i = 1, ..., N\}$$

Each E_i a subset of all possible inputs.

- Certain sets, E^e, produce an execution failure (exection terminates prematurely, fails to terminate, results produced unacceptable).
- Propability a program run will fail. $P = N^e/N$ where N^e is the number of E^e

- Reliability is probability of no failure.
  - $R = 1-p$

Workload dependencies characterized by a probability distribution

  - P^i: $i = 1, ..., N$
    $P = P^iY^i$ where $Y^i = 0$ if run with E^i successful

Reliability after n runs

  - $R(n) = R^n = (1-p)^n$

If runtimes known for each E^i, can calculate probability of no failure in a time interval.

Axiomatic Models

- Software follows certain universal laws
- Halsted provided quantitative measures of
  - program level
    language level
    algorithm purity
    program clarity
    effect of modularization
    programming effort
    programming time
- Bugs
  - $B = KV/EO$

where

K a proportionality constant

V the volume of the implementation

algorithm

EO the mean number of mental

discriminations between

errors made by programmers

$V = N \log_2 n$

N is program length

n is size of vocabulary

$N = N1 + N2$

N1 - total number of occurrences of the operators in a program

N2 - total number of occurrences of operands in a program

$n = n1 + n2$

n1 - number of distinct operators appearing in a program

n2 - number of distinct operands appearing in a program

EO empirically estimated to be about 3000

Reference	Correlation Coefficient Between Predicted and Real Number of Bugs
[Funami and Halstead, 1975]	0.98-0.83-0.92
[Cornell and Halstead, 1976]	0.99
[Fitzsimmons and Love, 1978]	System A 0.81
System 0.75	System C 0.75
	Total 0.76

Other Models

Costes, Landrault, Laprie

- Software error rate a function of both input data sets and logical paths.
- Semi Markov model with state occupancies and transition rates assumed known or measurable. Hydrid Models Using Measured Statistics

APPENDIX A: SOFTWARE ERROR CATEGORIES

From Brown and Buchanan [10]

A000		COMPUTATIONAL ERRORS
	A100	Incorrect operand in equation
	A200	Incorrect use of parenthesis
	A300	Sign convention error
	A400	Units or data conversion error
	A500	Computation produces an over/under flow
	A600	Incorrect/inaccurate equation used
	A700	Precision loss due to mixed mode
	A800	Missing computation
	A900	Rounding or truncation error
B000		LOGIC ERRORS
	B100	Incorrect operand in logical expression
	B200	Logic activities out of sequence
	B300	Wrong variable being checked
	B400	Missing logic or condition tests
	B500	Too many/few statements in loop
	B600	Loop iterated incorrect number of times (including endless loop)
	B700	Duplicate logic
C000		DATA INPUT ERRORS
	C100	Invalid input read from correct data file
	C200	Input read from incorrect data file
	C300	Incorrect input format
	C400	Incorrect format statement referenced
	C500	End of file encountered prematurely
	C600	End of file missing
D000		DATA HANDLING ERRORS
	D050	Data file not rewound before reading
	D100	Data initialization not done
	D200	Data initialization done improperly
	D300	Variable used as a flag or index not set properly
	D400	Variable referred to by the wrong name
	D500	Bit manipulation done incorrectly
	D600	Incorrect variable type
	D700	Data packing/unpacking error
	D800	Sort error
	D900	Subscripting error

APPENDIX A: SOFTWARE ERROR CATEGORIES (Continued

E000		DATA OUTPUT ERRORS
	E100	Data written on wrong file
	E200	Data written according to the wrong format statement
	E300	Data written in wrong format
	E400	Data written with wrong carriage control
	E500	Incomplete or missing output
	E600	Output field size too small
	E700	Line count or page eject problem
	E800	Output garbled or misleading
F000		INTERFACE ERRORS
	F100	Wrong subroutine called
	F200	Call to subroutine not made or made in wrong place
	F300	Subroutine arguments not consistent in type, units, order, etc.
	F400	Subroutine called is nonexistent
	F500	Software/data base interface error
	F600	Software/user interface error
	F700	Software/software interface error
G000		DATA DEFINITION ERRORS
	G100	Data not properly defined/dimensioned
	G200	Data referenced out of bounds
	G300	Data being referenced at incorrect location
	G400	Data pointers not incremented properly
H000		DATA BASE ERRORS
	H100	Data not initialized in data base
	H200	Data initialized to incorrect value
	H300	Data units are incorrect
I000		OPERATION ERRORS
	I100	Operating system error (vendor supplied)
	I200	Hardware error
	I300	Operator error
	I400	Test execution error
	I500	User misunderstanding/error
	I600	Configuration control error

APPENDIX A: SOFTWARE ERROR CATEGORIES (Continued)

J000		OTHER
	J100	Time limit exceeded
	J200	Core storage limit exceeded
	J300	Output line limit exceeded
	J400	Compilation error
	J500	Code or design inefficient/not necessary
	J600	User/programmer requested enhancement
	J700	Design nonresponsive to requirements
	J800	Code delivery or redelivery
	J900	Software not compatible with project standards
K000		DOCUMENTATION ERRORS
	K100	User manual
	K200	Interface specification
	K300	Design specification
	K400	Requirements specification
	K500	Test documentation
X0000		PROBLEM REPORT REJECTION
	X0001	No problem
	X0002	Void/withdrawn
	X0003	Out of scope - not part of approved design
	X0004	Duplicates another problem report
	X0005	Deferred

TABLE 5.3-1
ESTIMATED STAGE PROBABILITIES

PROGRAM	STAGE	PROB. OF ERROR PER EXECUTION
A1	1	.9803
	2	.1068
	3	.002602
	4	.002104
	5	.001176
	6	.0007659
B1	1	.1524
	2	.4673
	3	.4098
	4	.009117
	5	.003483
	6	.0005359
A2	1	.8065
	2	.2632
	3	.03759
	4	.01909
	5	.0001374
B2	1	.1449
	2	.02625
	3	.0001033
A3	1	.2488
	2	.05376
	3	.03524
	4	.02691
	5	.02302
B3	1	.3650
	2	.1037
	3	.01021
	4	.004681
	5	.002411

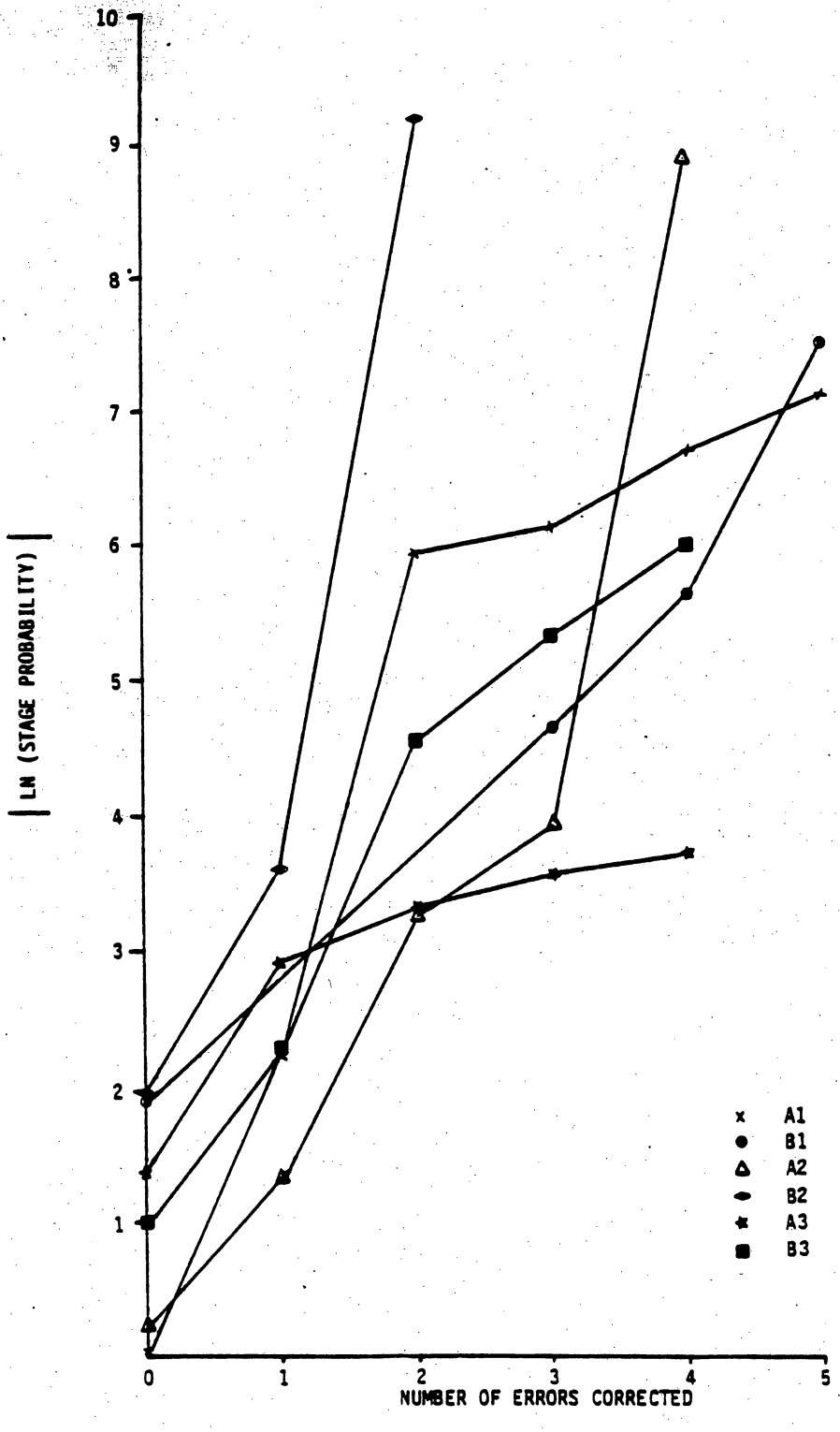


Figure 5.3-1. Estimated Error Rate as a Function of Errors Corrected, Using Original Data.

TABLE 5.5-3
ESTIMATES FOR ERROR PROBABILITY PER PROGRAM EXECUTION, BASED ON THE PROPORTIONAL HAZARDS MODEL

PROGRAM	STAGE	EXPERIMENT BASED ESTIMATES	MODEL PREDICTIONS (ORIGINAL ORIGIN)	MODEL PREDICTIONS (MODIFIED ORIGIN)
A1	1	.9803	.4197	
	2	.1068	.05678	.04430
	3	.002602	.007689	.007936
	4	.002104	.001041	.001422
	5	.001176	.0001409	.0002546
	6	.0007659	.00001907	.00004561
B1	1	.1524	.1829	.0812
	4	.009117	.006947	.005348
	5	.003483	.002335	.002160
	6	.0005359	.007850	.0008721
A2	1	.8065	1.3142*	
	2	.2632	.2054	.1770
	3	.03759	.03209	.02230
	4	.01909	.005015	.002809
	5	.0001374	.0007836	.0003538
B2	1	.1449	.3312	.3241
	2	.02625	.007486	.007411
	3	.0001033	.0001692	.0001694
A3	1	.2488	.2487	.1529
	2	.05376	.1117	.08417
	3	.03524	.05109	.04633
	4	.02691	.02255	.02550
	5	.02302	.01012	.01404
B3	1	.3650	.1083	.2800
	2	.1037	.04864	.06904
	3	.01021	.02185	.01702
	4	.004681	.009816	.004198
	5	.002411	.004410	.001035

* Unconstrained Model

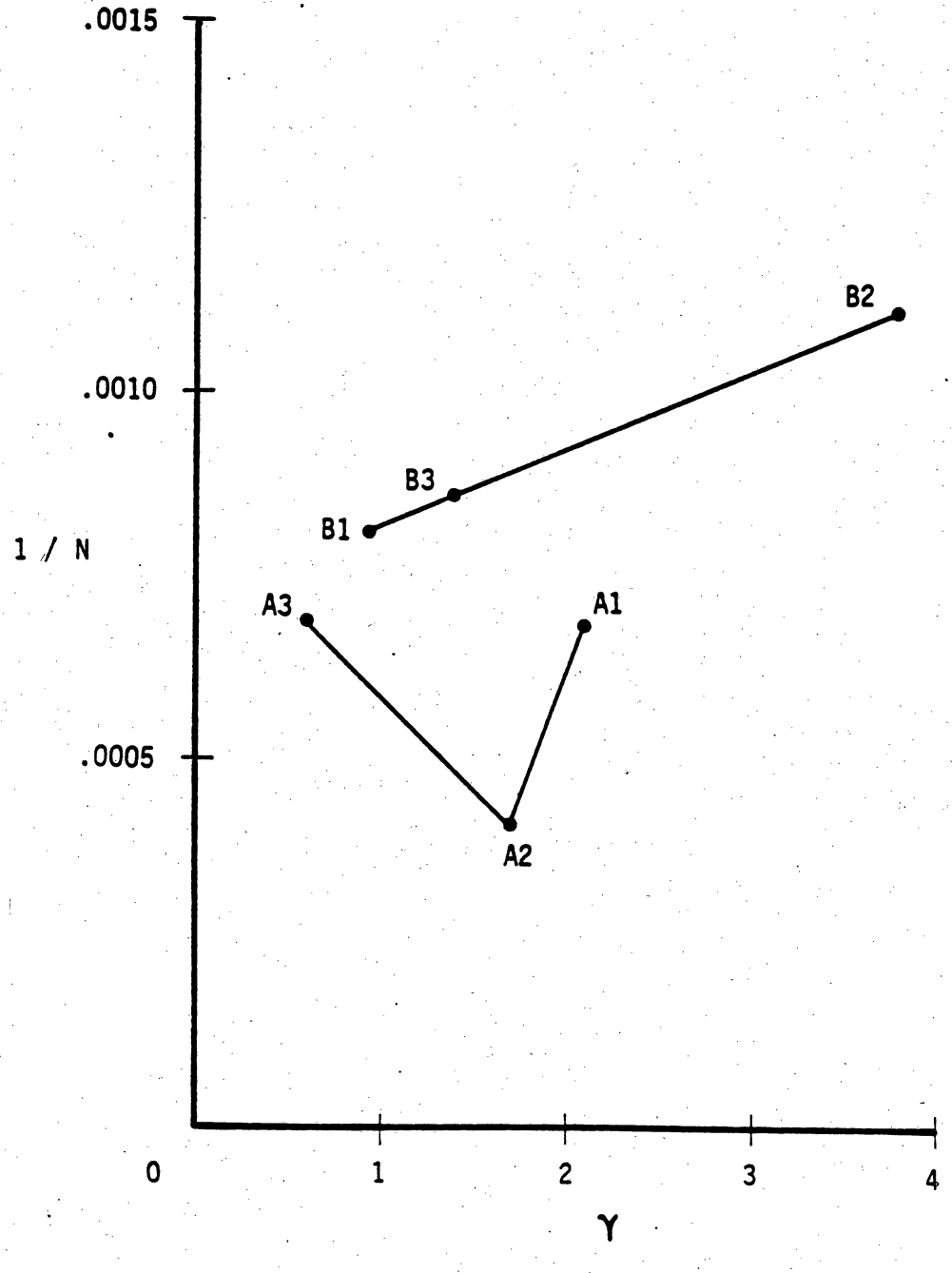


Figure 5.6-1. Inverse of Halstead's Length vs. Predicted Slope (Modified Data), Proportional Hazards Model.

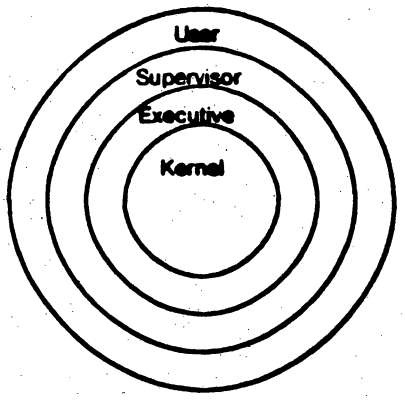
Table 4. The three-version decision algorithm, where G = Good Point, D = Detected Error, and U or U* = Undetected Error (either distinct or similar, respectively). The confidence level describes how many version results the decision algorithm used in obtaining the decision result.

CASE	VERSION RESULTS	NO. OF ERRORS	DECISION ALGORITHM	DECISION RESULT	CONF. LEVEL
V1	G,G,G	0	TRIPLEX	G	3
V2	G,G,D	1	DUPLEX	G	2
V3	G,G,U	1	TRIPLEX	G	2
V4	G,D,D	2	SIMPLEX	G	1
V5	G,D,U	2	DUPLEX	D	0
V6	G,U,U	2	TRIPLEX	D	0
V7	G,U*,U*	2	TRIPLEX	U*	2
V8	D,D,D	3	NULL	D	0
V9	D,D,U	3	SIMPLEX	U	1
V10	D,U,U	3	DUPLEX	D	0
V11	D,U*,U*	3	DUPLEX	U*	2
V12	U,U,U	3	TRIPLEX	D	0
V13	U,U*,U*	3	TRIPLEX	U*	2
V14	U*,U*,U*	3	TRIPLEX	U*	3

Table 5. The results of decision algorithms.

CASE	ALL	THREE-VERSION TRIPLET COMPOSITION				
		000	PPP	EEE	OPE	OTHER
V1	36665	1703	448	820	9354	24340
	44.9%	48.7%	44.8%	41.0%	44.5%	45.0%
V2	5292	129	128	166	1341	3528
	6.5%	3.7%	12.8%	8.3%	6.4%	6.5%
V3	22105	842	274	554	5939	14496
	27.1%	24.1%	27.4%	27.7%	28.3%	26.8%
V4	1283	48	8	32	347	848
	1.6%	1.4%	0.8%	1.6%	1.7%	1.6%
V5	3986	141	88	80	1046	2631
	4.9%	4.0%	8.8%	4.0%	5.0%	4.9%
V6	6838	264	18	206	1747	4603
	8.4%	7.5%	1.8%	10.3%	8.3%	8.5%
V7	1944	86	12	82	386	1378
	2.4%	2.5%	1.2%	4.1%	1.8%	2.5%
V8	176	4	0	0	65	107
	0.2%	0.1%	0.0%	0.0%	0.3%	0.2%
V9	477	20	7	4	123	323
	0.6%	0.6%	0.7%	0.2%	0.6%	0.6%
V10	867	11	6	22	274	554
	1.1%	0.3%	0.6%	1.1%	1.3%	1.0%
V11	87	6	0	0	28	53
	0.1%	0.2%	0.0%	0.0%	0.1%	0.1%
V12	1415	173	1	20	275	946
	1.7%	4.9%	0.1%	1.0%	1.3%	1.7%
V13	353	48	0	8	62	235
	0.4%	1.4%	0.0%	0.4%	0.3%	0.4%
V14	112	25	10	6	13	58
	0.1%	0.7%	1.0%	0.3%	0.1%	0.1%
Total	81600	3500	1000	2000	21000	54100
	100.0%	100.0%	100.0%	100.0%	100.0%	100.0%

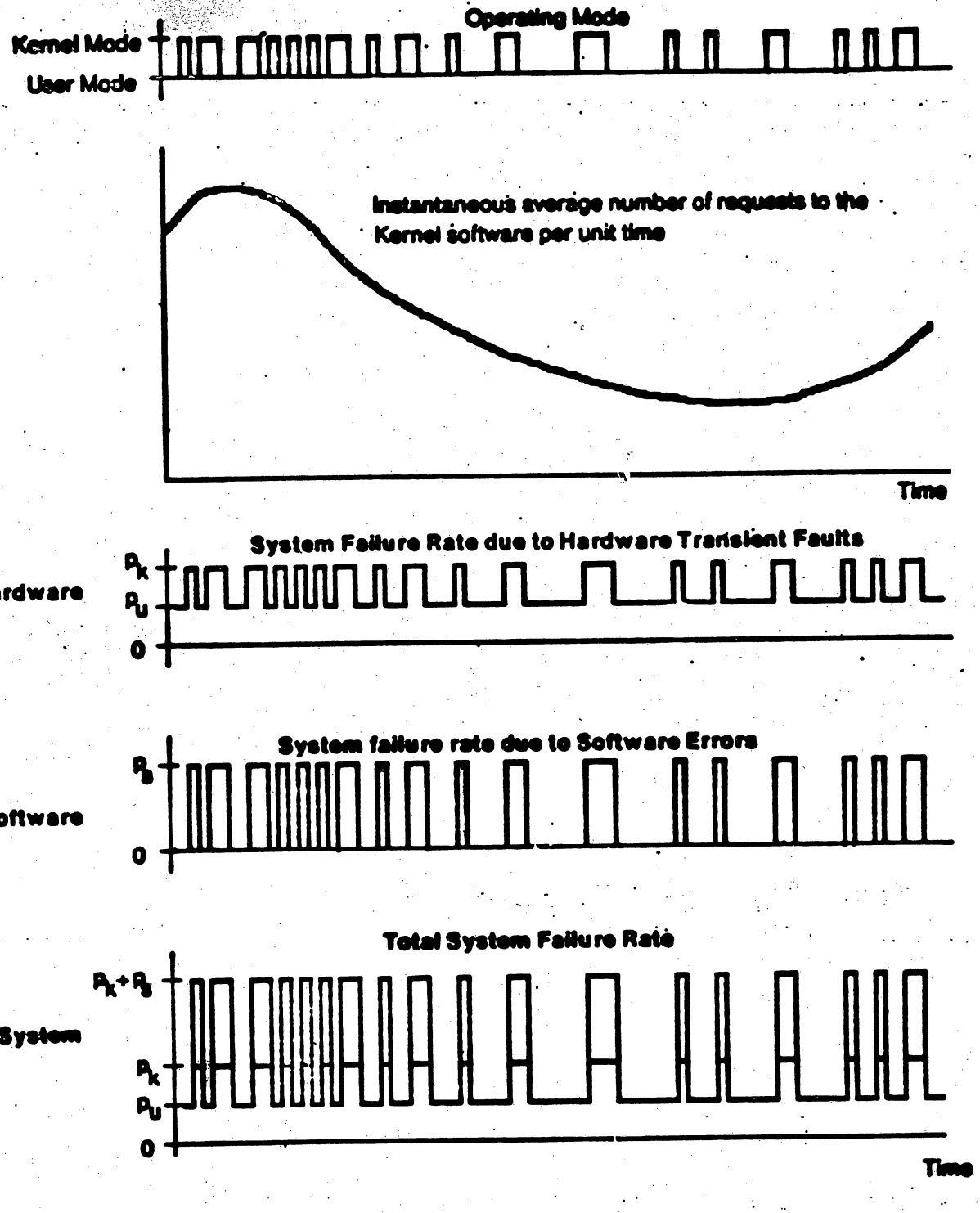
Validation with a MULTICS like TSS



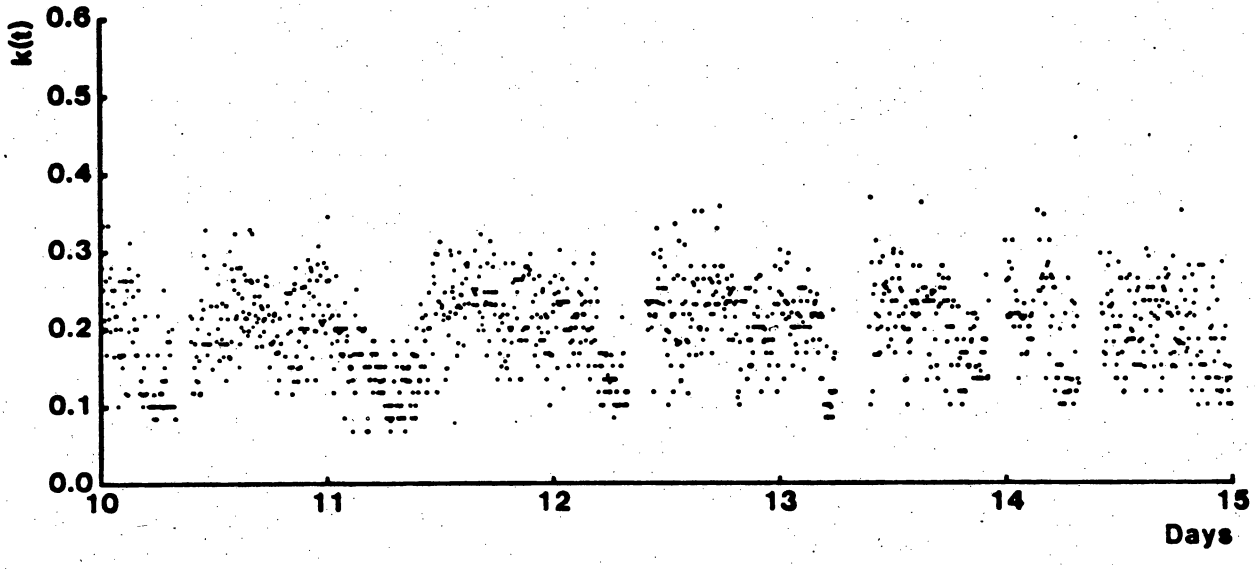
Hypothesis

- Software faults in the kernel are *more likely* to lead to a system failure than software faults in any of the outer rings.
- Transient hardware faults affecting the kernel code or data structures are *more likely* to lead to a system failure than transients affecting programs in the outer rings (recovery is not attempted at all in 4341 processors for errors detected in kernel mode).
- Therefore, the observed system reliability will depend on kernel usage.

The underlying failure process



Validation



Issues :

- Estimate periodic component
- Estimate autocorrelation function of random component

or positive failure rate

od estimation of model

delete tex

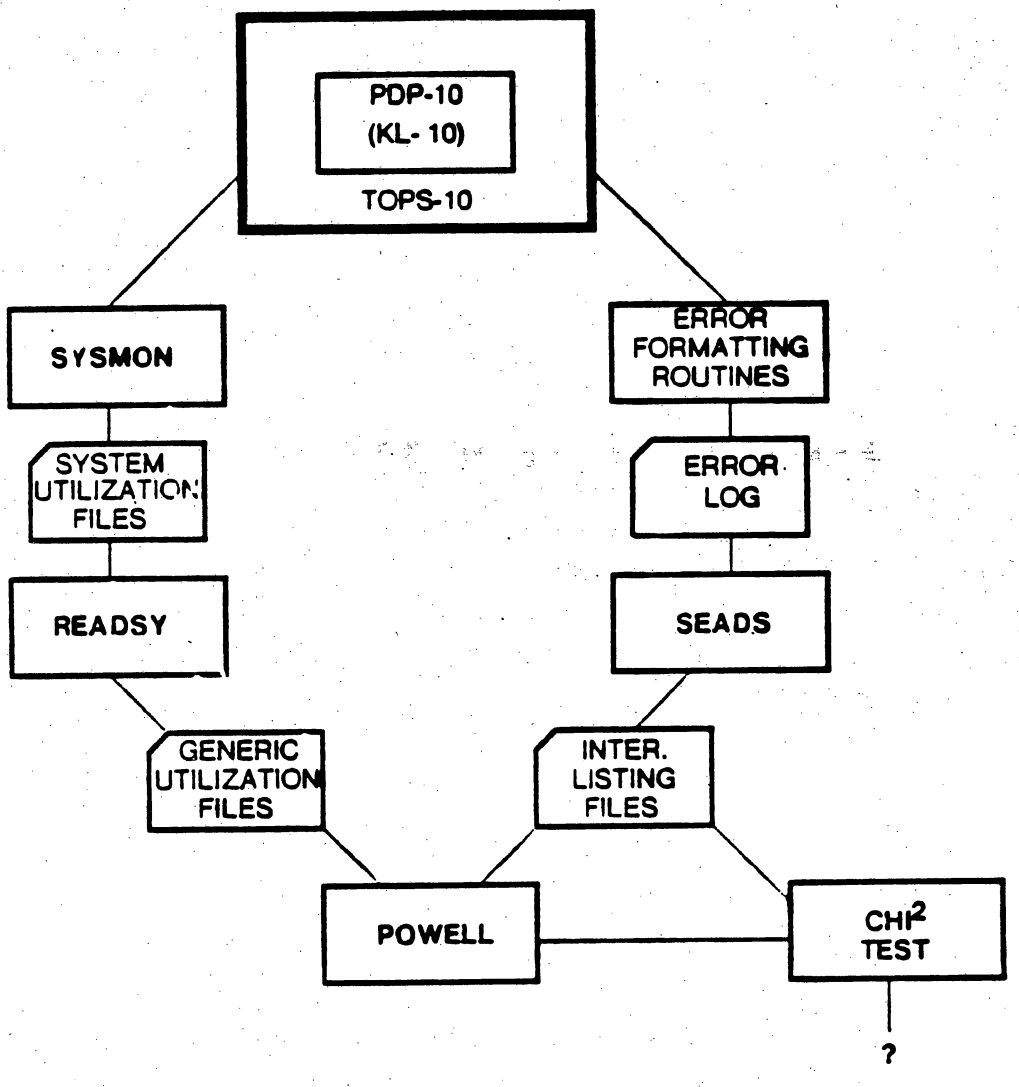
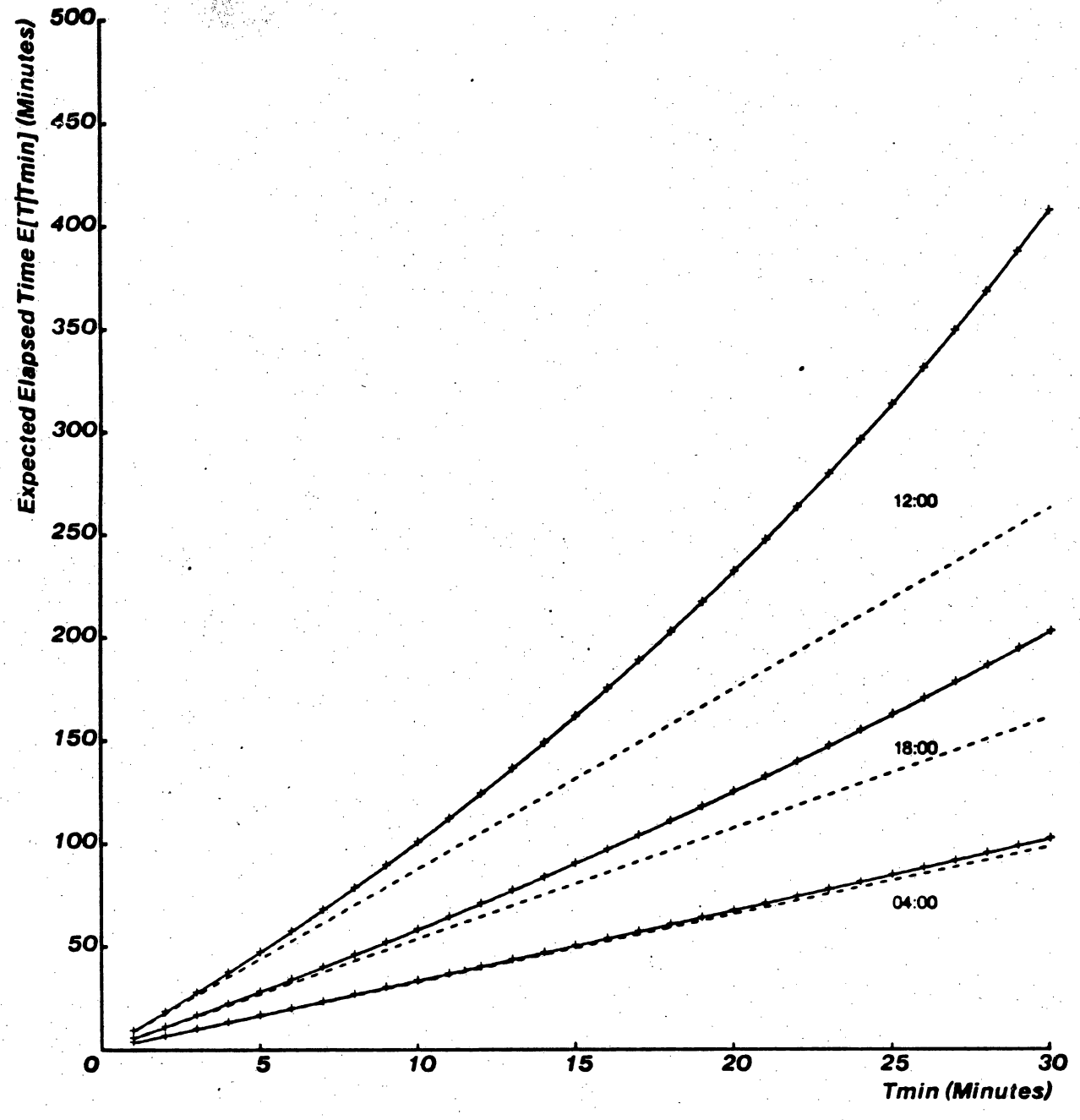
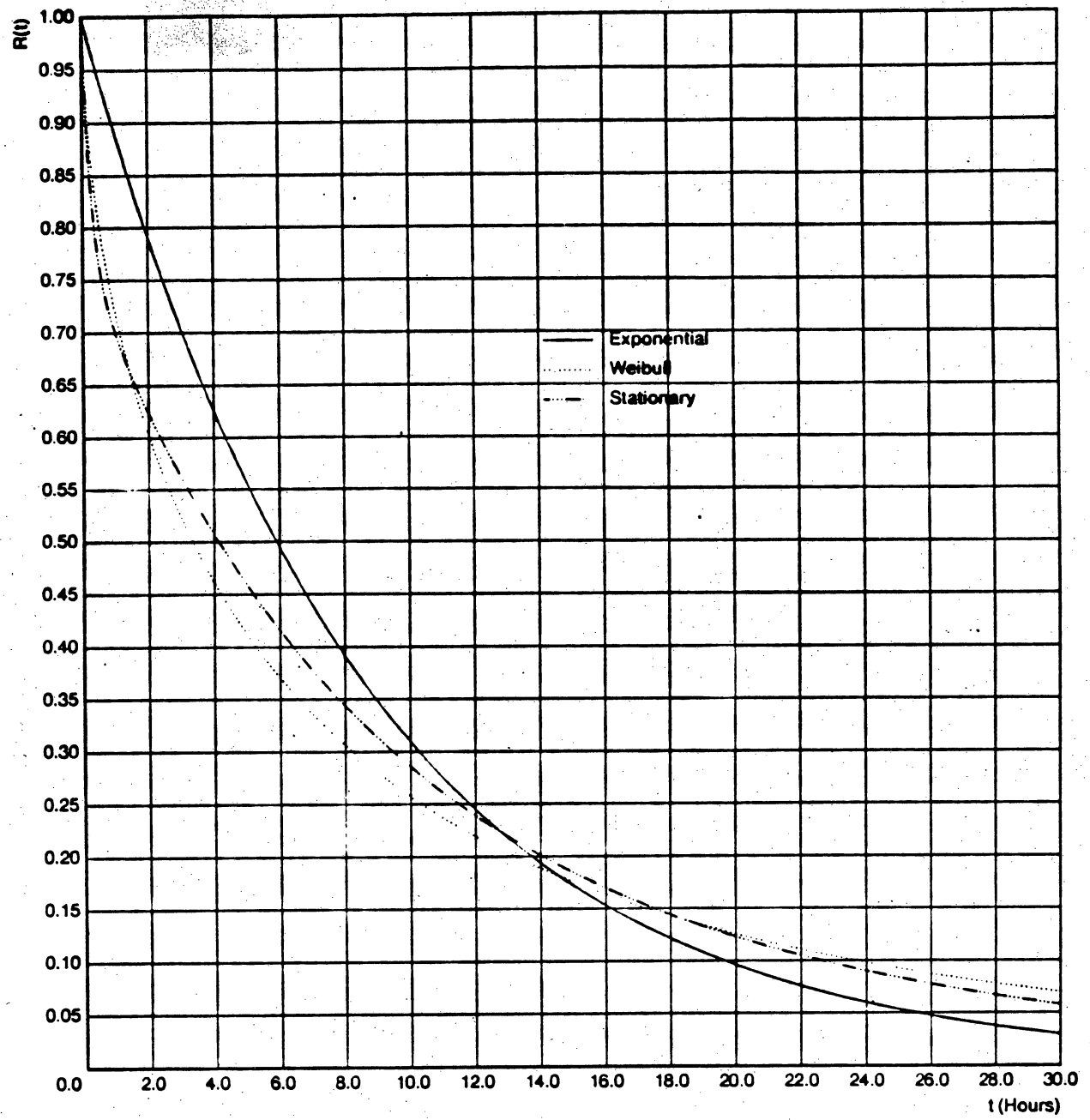


Figure 3-1: Software packages used in the validation of the cyclostationary modeling methodology.


Expected Elapsed Time E[T|Tmin] (Minutes)
500
450
400
350
300
250
200
150
100
50
0
5
10
15
20
25
30
Tmin (Minutes)
12:00
18:00
04:00


R(t)
1.00
0.95
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0.55
0.50
0.45
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.0
2.0
4.0
6.0
8.0
10.0
12.0
14.0
16.0
18.0
20.0
22.0
24.0
26.0
28.0
30.0
t (Hours)
Exponential
Weibull
Stationary

Exponential

$$R_e(\tau) = e^{-\lambda_e \tau} \tag{4.8}$$

$$h_e(\tau) = \lambda_e \tag{4.9}$$

Weibull

$$R_w(\tau) = e^{-(\lambda_w \tau)^{\alpha_w}} \tag{4.10}$$

$$h_w(\tau) = \frac{\alpha_w \lambda_w}{(\lambda_w t)^{1-\alpha_w}} \tag{4.11}$$

Periodic

$$R_p(\tau) = e^{-\lambda_p \tau} e^{-F_p u(\tau)} \tag{4.12}$$

$$h_p(\tau) = \left[\lambda_p + F_p \frac{\partial u(\tau)}{\partial \tau} \right] \tag{4.13}$$

Cyclostationary

$$R_c(\tau) = e^{-(\lambda_c + \sigma_{c1} + \sigma_{c2})\tau - \frac{\sigma_{c1}}{\beta_1}[1 - e^{-\beta_1 \tau}] - \frac{\sigma_{c2}}{\beta_2}[1 - e^{-\beta_2 \tau}] + \ln \phi(t)} \tag{4.14}$$

$$h_c(\tau) = \lambda_c - \sigma_{c1}[1-e^{-\beta_1 \tau}] - \sigma_{c2}[1-e^{-\beta_2 \tau}] - \frac{1}{\phi(t)} \frac{\partial \phi(t)}{\partial t} \tag{4.15}$$

Simplified Cyclostationary

$$R_m(\tau) = e^{-(\alpha_m - \gamma_m)\tau - \frac{\gamma_m}{\beta_m}[1 - e^{-\beta_m \tau}]} \tag{4.16}$$

$$h_m(\tau) = \alpha_m - \gamma_m[1 - e^{-\beta_m \tau}] \tag{4.17}$$

Table 4-1: Reliability and Hazard functions of the five compared models.

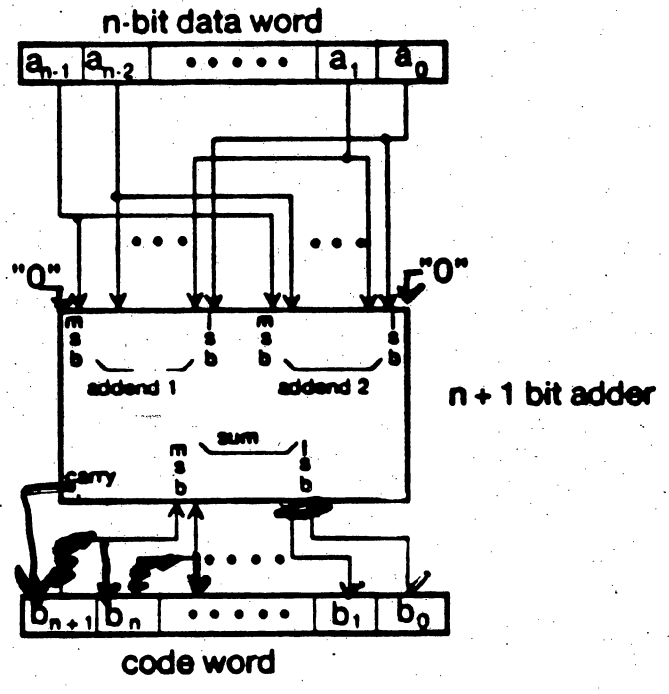
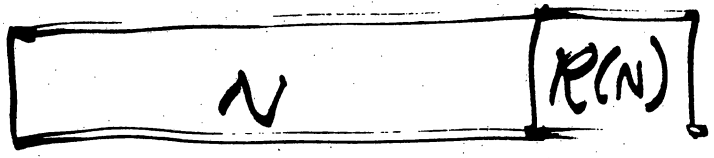


Figure <AN coder> Simple Encoder for 3N Single Error Detecting Arithmetic Code

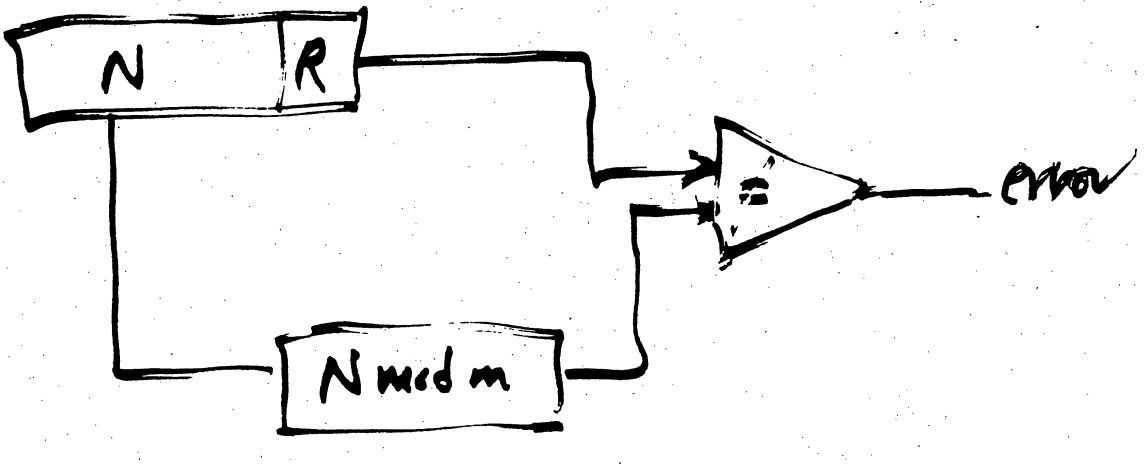
Residue Code

Residue - m

$R(N) = N \bmod m$



separable



CYCLIC CODES

(n,k) cyclic codes can detect bursts $\leq$ [redacted] (n-k) bits in length.

(n,k) = (r + k,k), where r is # of check bits

Cyclic Codes uniquely specified by a Generator Polynomial

$$G(x) = X^r + a_{r-1}X^{r-1} \ldots + a_1X + a_o$$

a_i = (0,1) for binary codes

Cyclic codes normally done in serial in a linear feedback shift register, but can be done in parallel (e.g. $\exists$ decoder matrix). In serial version, though, data length k is arbitrarily long, albeit usually a multiple of character length.

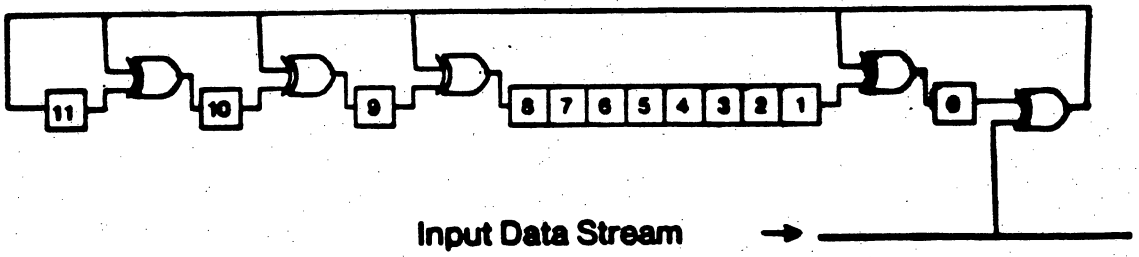
Example G(x)'s

CRC-12 $X^{12} + X^{11} + X^3 + X^2 + X + 1$

CRC-16 $X^{16} + X^{15} + X^2 + 1$

CRC-CCITT $X^{16} + X^{12} + X^5 + 1$

Detects bursts $\leq$ 16 bits
99% of bursts > 16 bits
ANSI X.25



Generator Polynomial $G(X) = X^{12} + X^{11} + X^3 + X^2 + X + 1$

Figure <CRC-12> Block Check Register for CRC-12 Cyclic Code

Place XOR at output of cell i

for $i = r - j$

$r = 12$, $j = 13, 11, 3, 2, 1$

$(12+k, k)$ code

Detects bursts of length 12 or less

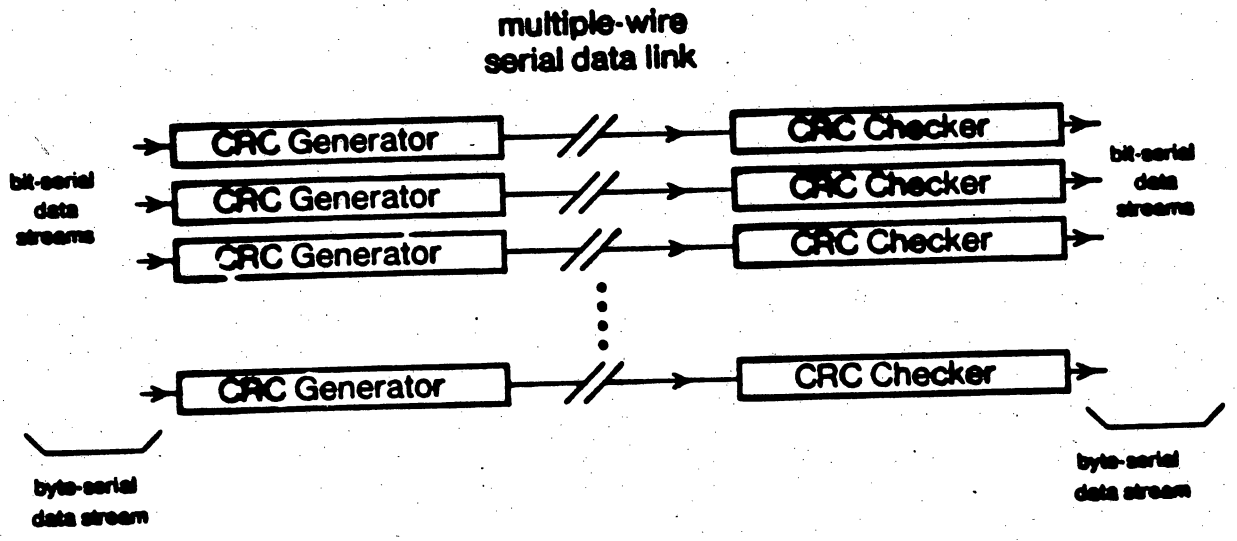


Figure <Multiple CRC> Use of cyclic codes for a bus which transfers data in a byte-serial fashion, i.e., the bits of each byte are transmitted simultaneously. The CRC check bits are generated for each bit stream during the duration of the block transmission and are appended at the end of the block.

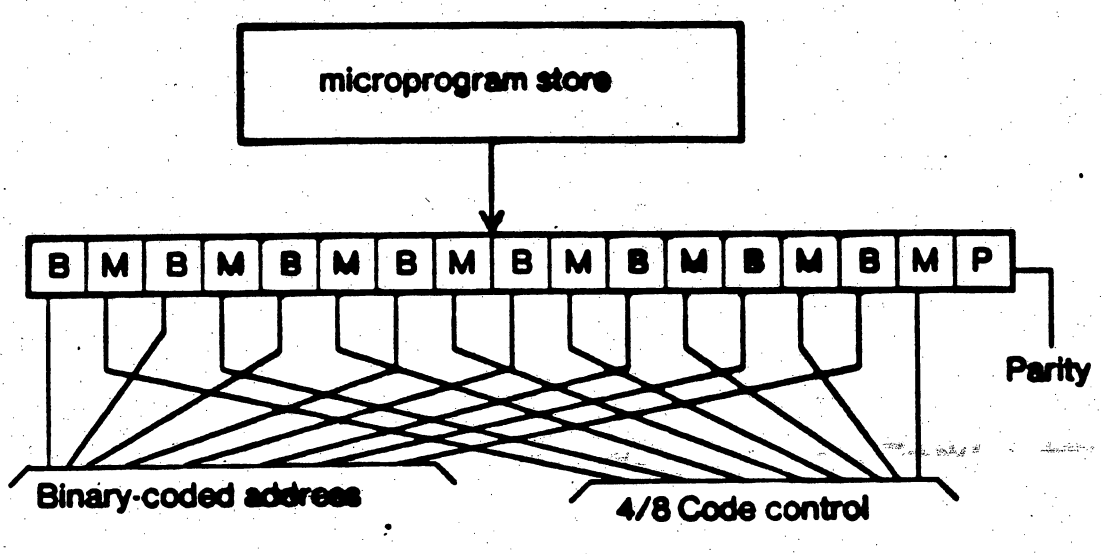
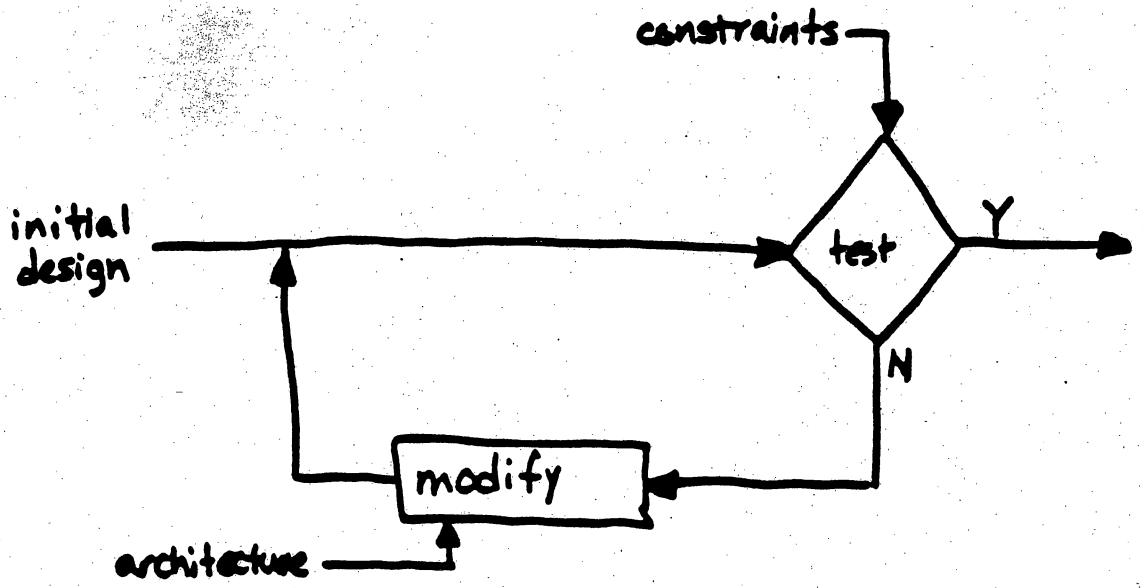
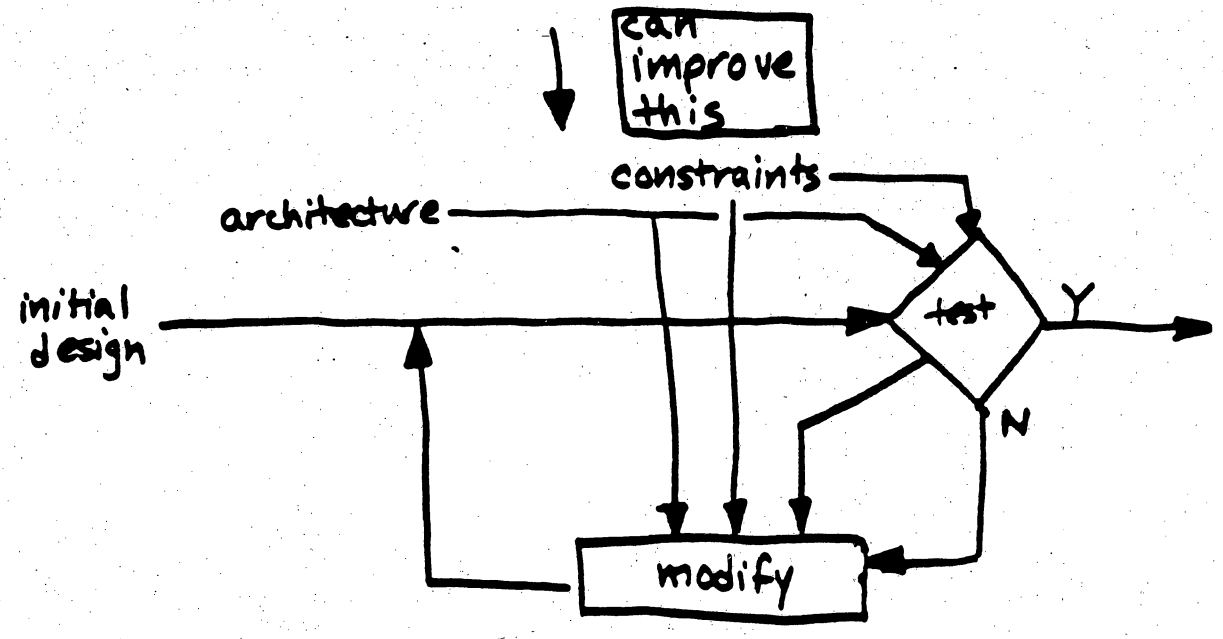


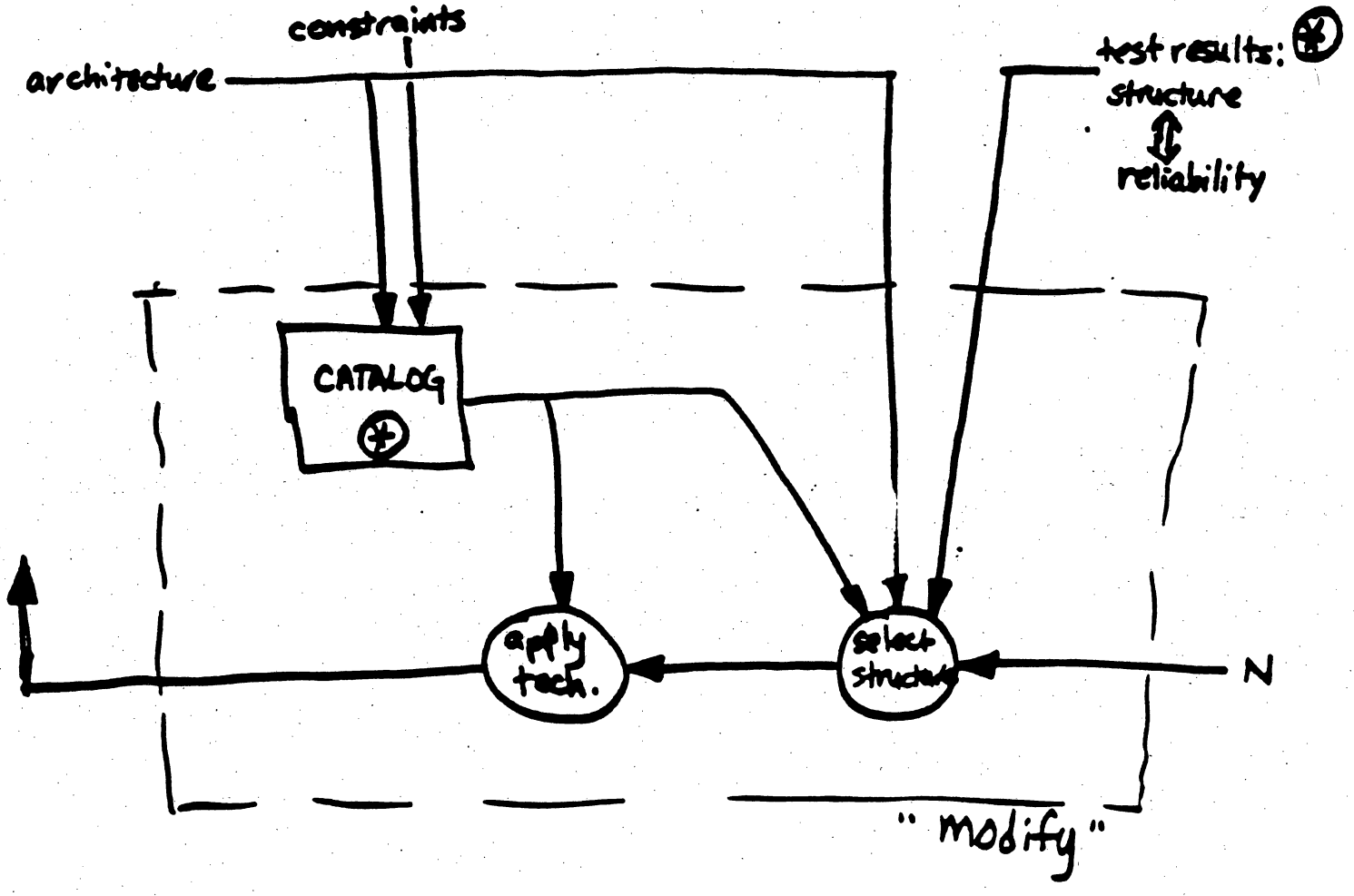
Figure (ESS-3A) 4/8 coding in Bell ESS-3A microstore. The control bits are 4/8 coded, and interlaced with the address bits. This scheme protects the address bits as well as the control bits from multiple adjacent unidirectional errors. (Source [TOY 79])

3.4) System's Design



As done today:
Key point: yes/no test




constraints
architecture
test results:
structure
reliability
CATALOG
apply tech.
select structure
N
"modify"

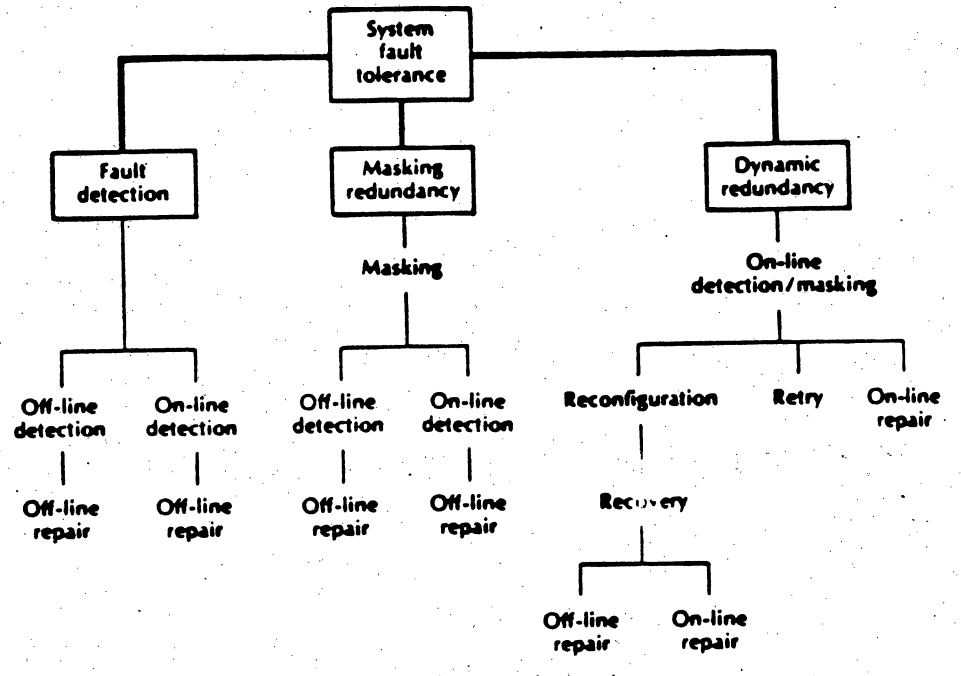
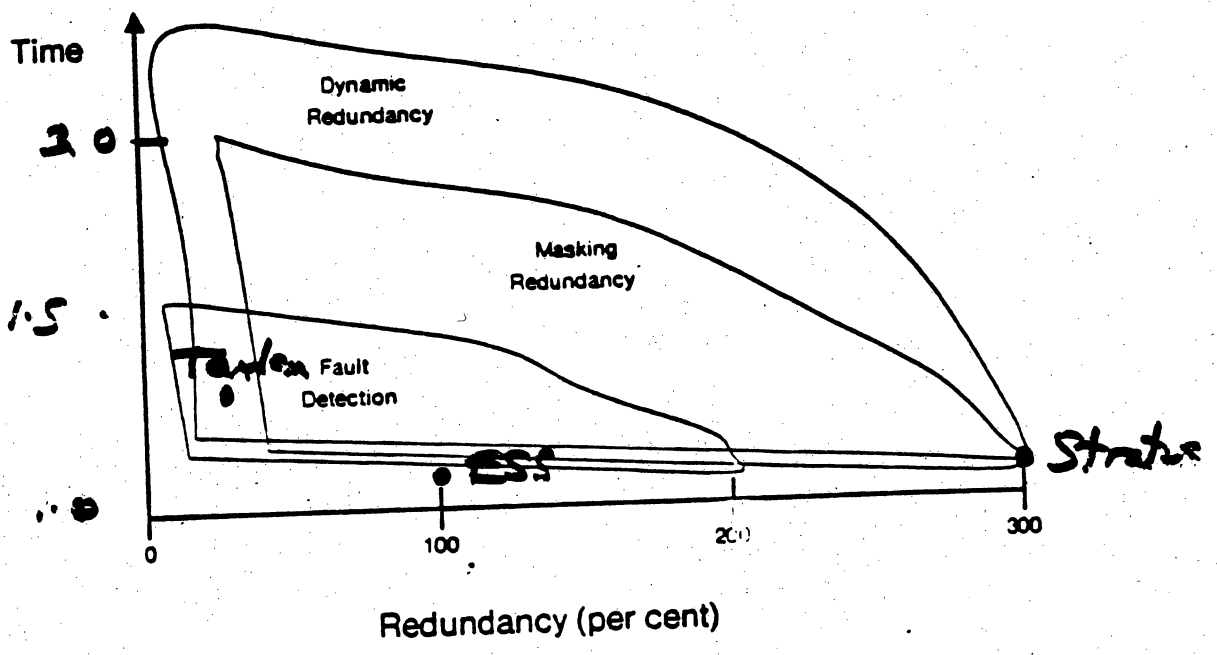


Figure 3–2. Taxonomy of system fault-tolerance strategies.


Time
Dynamic
Redundancy
Masking
Redundancy
Fault
Detection
0
100
300
Redundancy (per cent)

The previous code had minimum distance of 2

- it's duplication, or a variant
- unlike straight duplication has coverage of all unidirectional errors
- distance not enough

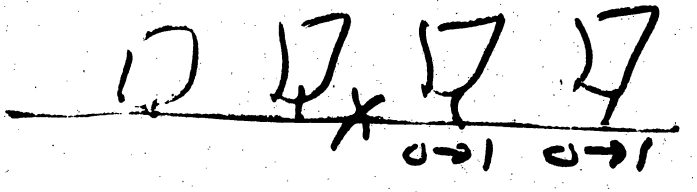
aim is code efficiencies:

1. minimum redundancy
2. maximum coverage, distance
3. minimum coding/detection/decoding hardware, time

Separable codes
(n,k)

Properties of Codes

Code	Bits in word	Code Words	Distance	Single Bit	Double Bit	0's	1's	Multiple Adjacent Unidir. Errors
CD	4	4	2	✓	66%	✓	✓	✓
3/4	4	6	2	✓	33%	✓	✓	✓
Even Parity	4	8	2	✓	0	✗	✗	✗
Odd Parity	1	8	2	✓	0	✓	✓	✗



Error detection codes

A systematic application of redundancy to information

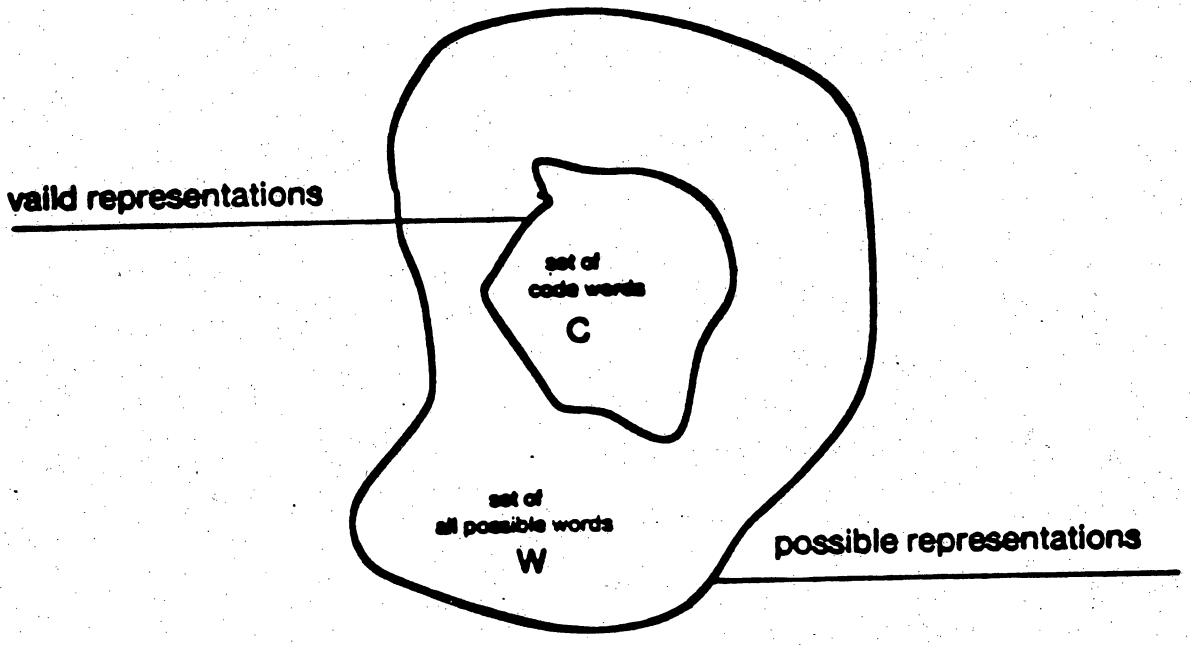


Figure <code space> The set of invalid representations (i.e. non-code words) is W-C

Measurement:

- Hamming distance (sometimes)
- Coverage
- Redundancy (& cost)

usually, both are needed

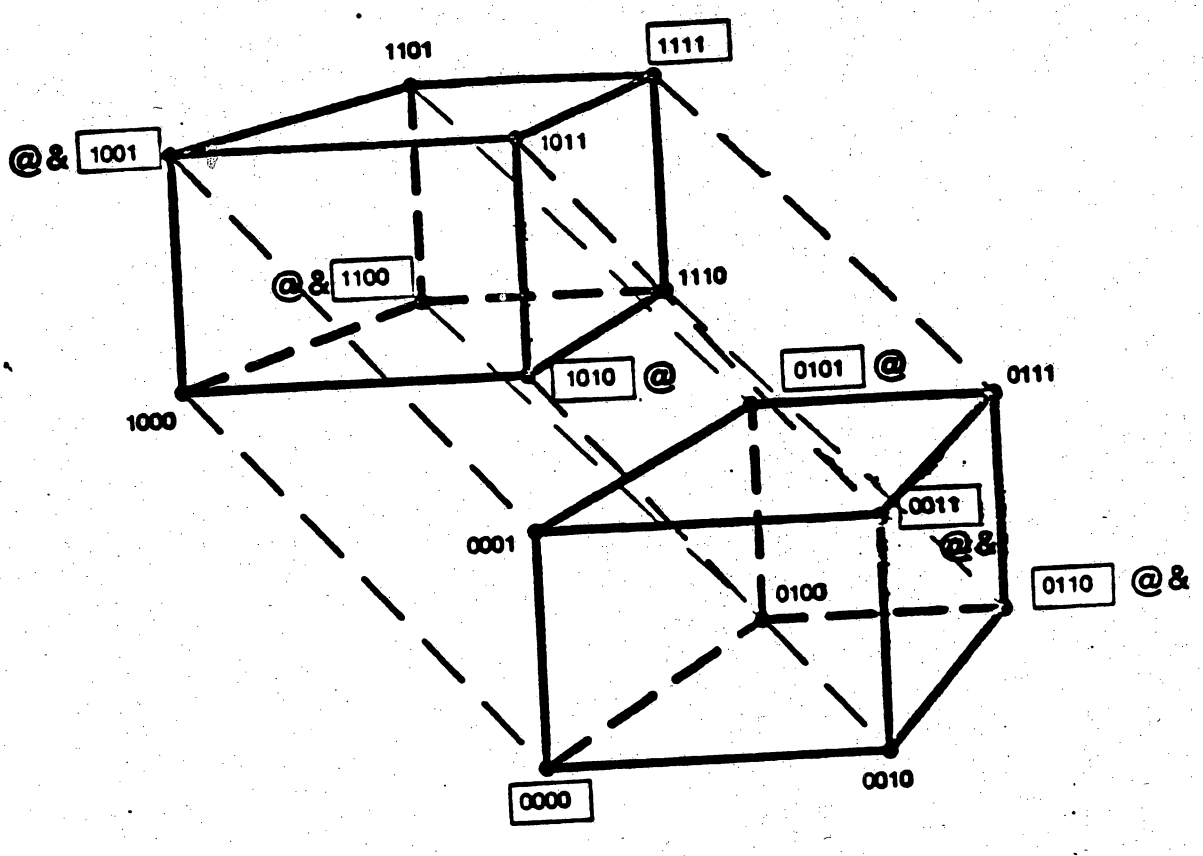
Catalog of Reliability Enhancement Techniques (cont.)

Example from Fault Avoidance Area: Summary of PDP-8/e Fault Avoidance Designs

Temp. (^{o}C)	π_Q	RAM π_Q	λ(*fpmh*)	MTTF (hours)	Notes
45	13	13	184.55	5419	Base design, convection cooling, plastic packages
40	13	13	122.75	8147	Fans installed plastic packages
35	13	13	78.10	12804	Cabinet ventilation system plastic packages
45	13	13	47.59	21013	Base design, convection cooling, ceramic packages
40	13	13	44.23	22609	Fans installed, ceramic memory chip packages only
45	13	8	125.73	7953	Base design, convection cooling, plastic packages
45	8	8	113.57	8805	Base design, convection cooling, plastic packages
40	13	8	37.23	26858	Fans installed, ceramic memory chip packages only
45	13	13	165.23	6052	Bit slice design, convection cooling, plastic packages
45	13	8	69.45	14398	Bit slice design, fans installed, plastic packages

Catalog of Reliability Enhancement Techniques

Region	Technique
Fault Avoidance	Environment Modification Quality Changes Component Integration Level
Fault Detection	Duplication Error Detection Codes › M-of-N codes › Parity Self-checking and Fail-safe logic Watch-dog timers and Timeouts Consistency and Capability checks
Masking Redundancy	NMR/voting Error correcting codes › Hamming SEC DED › Other codes Masking Logic › Interwoven Logic › Coded-state Machines
Dynamic Redundancy	Reconfigurable Duplication Reconfigurable NMR Backup sparing Graceful degradation



Figure<distcubebigger> Expanded cube made by adding one more bit to the word size. Boxed words are even parity words, @ marks a code word in a 2/4 m-of-n code, and & marks code words from the complemented duplication code used as an example in the text. The unmarked words are odd parity code words.

Application dependencies:

1. Decoding, etc. serial
   parallel
   speed
2. Code invariance to data operation
3. Fault types prevailing

Codes can be modified to meet constraints.

M-of-N CODES

N bits/word $(a_o a_1 \ldots a_{n-1})$

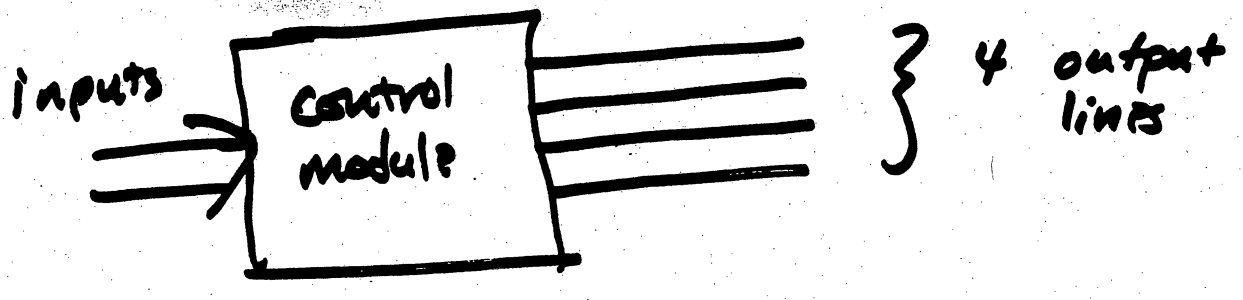
$$\sum_{0}^{N-1} a_i = M$$

- single & unidirectional faults.
- complex decoding unless serial
- simple
- large amount of redundancy need ${}_nC_m$ code words

Common use - control circuitry
3/5 ex. (15)

Bell ESS-3a 4/8

M/N EXAMPLE



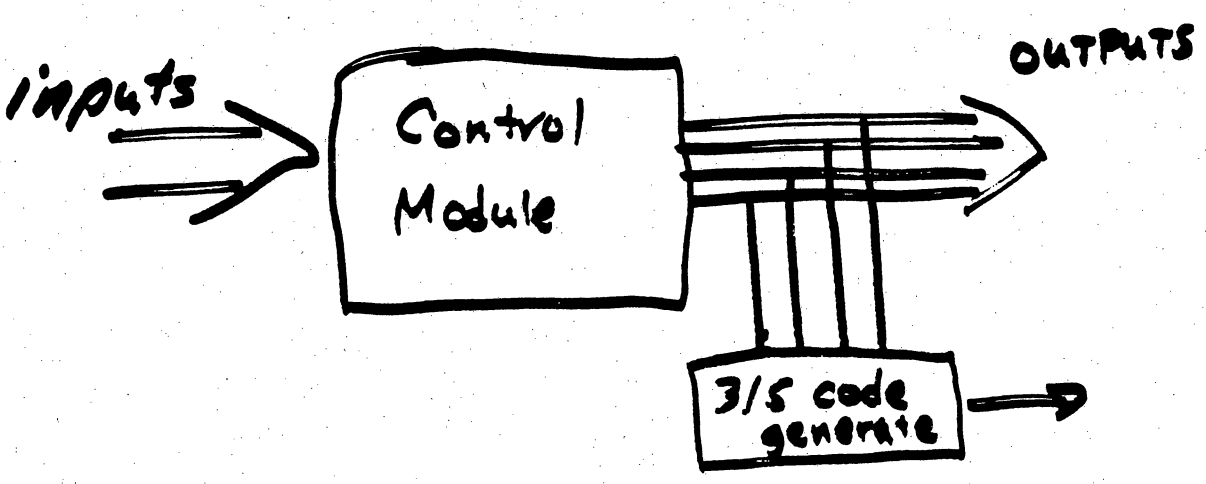
possible outputs @ any one time

1001	1011
1010	1110
1100	1101
0110	0111

Note that $3 \geq n \geq 2$ lines set at any one time.

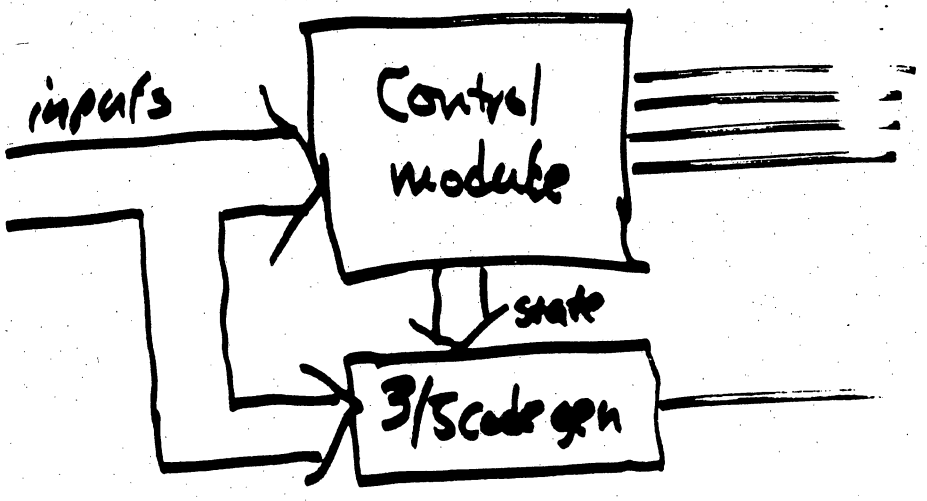
use an extra line to give a separable

3/5 code


inputs
Control Module
OUTPUTS
3/5 code generate

No! Does not check module function!

Instead:



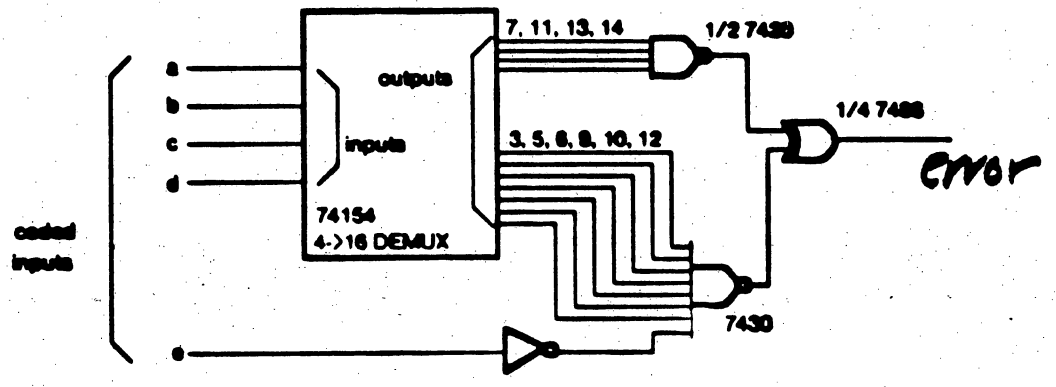


Figure <3/5 detect> Simple 3/5 code error detector

PARITY CODES

- Linear separable codes
- Parity - odd or even
- Generation by b - bit XOR gate or parity tree
  $\lceil \log_2 b \rceil$ gate levels
- Decoding uses b + 1 - bit tree
- Bit/word
- Bit/byte
  - varying parity sense
    - C.mmp shared memory
- Interlaced parity
- Chip wide parity (special case of interlaced parity)

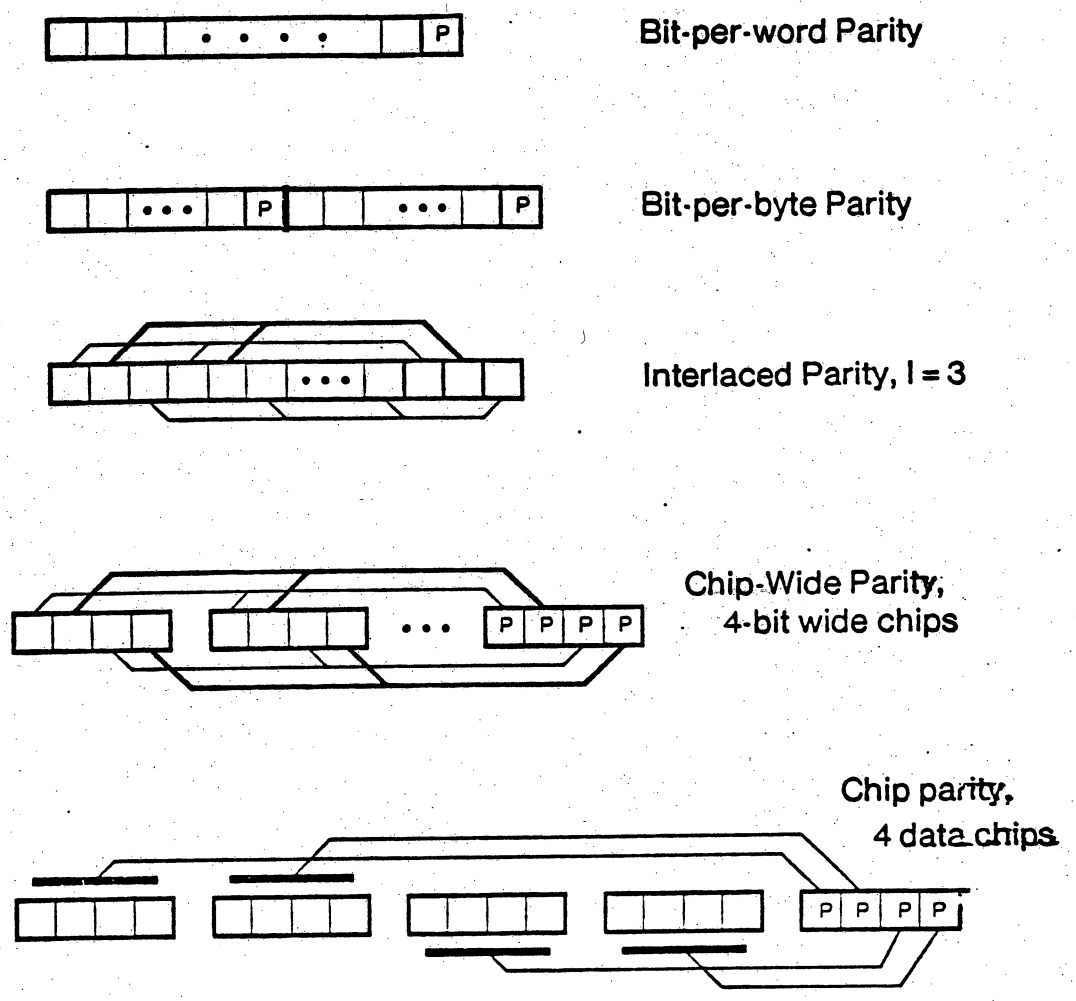


Figure <parity schemes> Examples of five different parity schemes.

	red	tree for decode	perf	ring reg
			$\lceil \log_2 b+1 \rceil$	word
BPW	$1/b$	$b+1$		
BPB	$1/m$	$\frac{b}{m}(n+1)$	$\lceil \log_2 m+1 \rceil$	byte
IPAR	i/b	$(b/i)+1$	$\lceil \log_2(\frac{b}{i}+1) \rceil$	group

m bits/byte
i parity groups

Table 3–8. Percentage of coverage of processor-memory bus failures.

	Coverage		
Error Type	(17,16) *Parity*	(17,16) *Cumulative*	(18,16) *Cumulative*
Hard failure:			
Bus all 1	50%	50%	100%
Bus all 0	50	50	100
Bus half 1	0	0	Near 100
Bus half 0	0	0	Near 100
Wire-or:			
2 wires	100	100	100
3 wires	0	0	88
4 wires	0	0	100
5 wires	0	0	0
Single bit*	100	100	100
Double bit			
Adjacent	0	0	100
Random	0	0	Near 100
Triple bit	100	100	100
Quadruple bit			
Two pairs adjacent	0	0	0
Two adjacent	0	0	Near 100
Three adjacent	0	0	100
Four adjacent	0	0	50
Random	0	0	0
Parity generate and check			
Stuck-at-ok	0	100	100
Stuck-at-1	50	50	Near 100
Stuck-at-0	50	50	Near 100

*One bit value, not a failed wire

time. The stored checksum is normally kept physically separate from the data block to limit the effect of a catastrophic failure on the fault-detecting capability.

Although checksumming is inexpensive in terms of excess information, it has three disadvantages. First, it is best suited to applications in which data are handled in large, contiguous blocks, such as buses that carry data in blocks, sequential storage, and block-transfer peripherals.

Second, checksumming in memories takes a long time to detect faults even when reading a single word, for s words must be read and added.

CHECK SUMS

- Cheap
- Performed over a block of 5 words
- Contiguous blocks - block Xfer devices best, since updating awkward - busses, tapes, disks .....
- Low diagnostic resolution

Four types

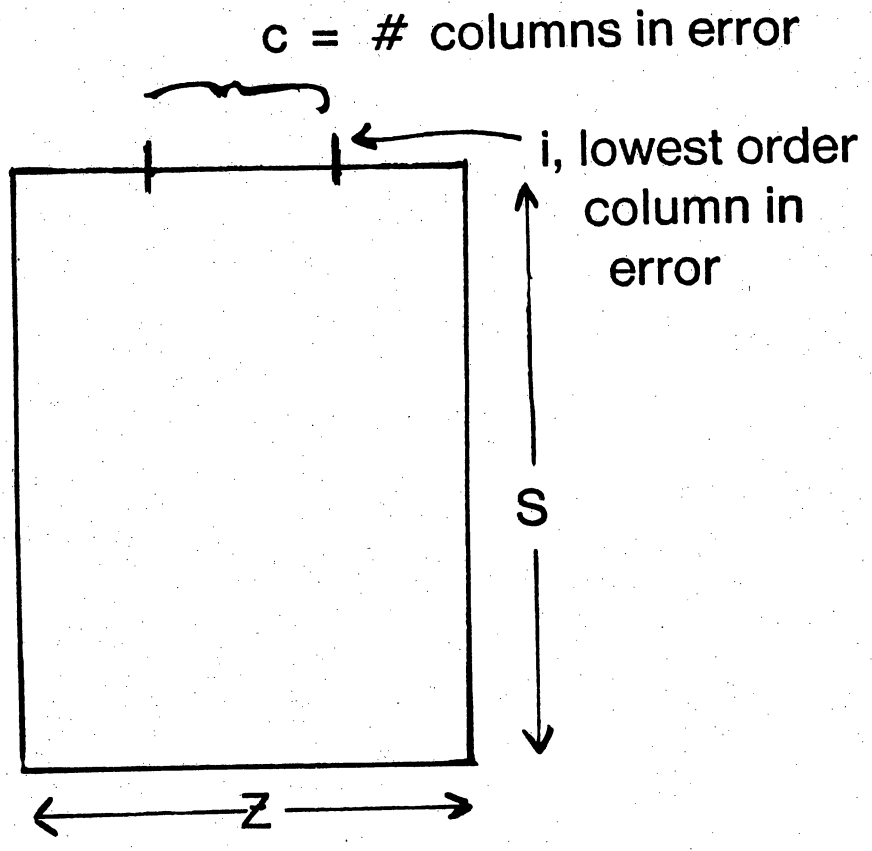
single precision: Z bits/word module 2^z addition

- error coverage varies with bit position - best in LSB

extend precision - module $2^z + \leftarrow\leftarrow$ addition

- extra bits in check sum store
- $S<2^a$ coverage same for all columns

CHECK SUMS



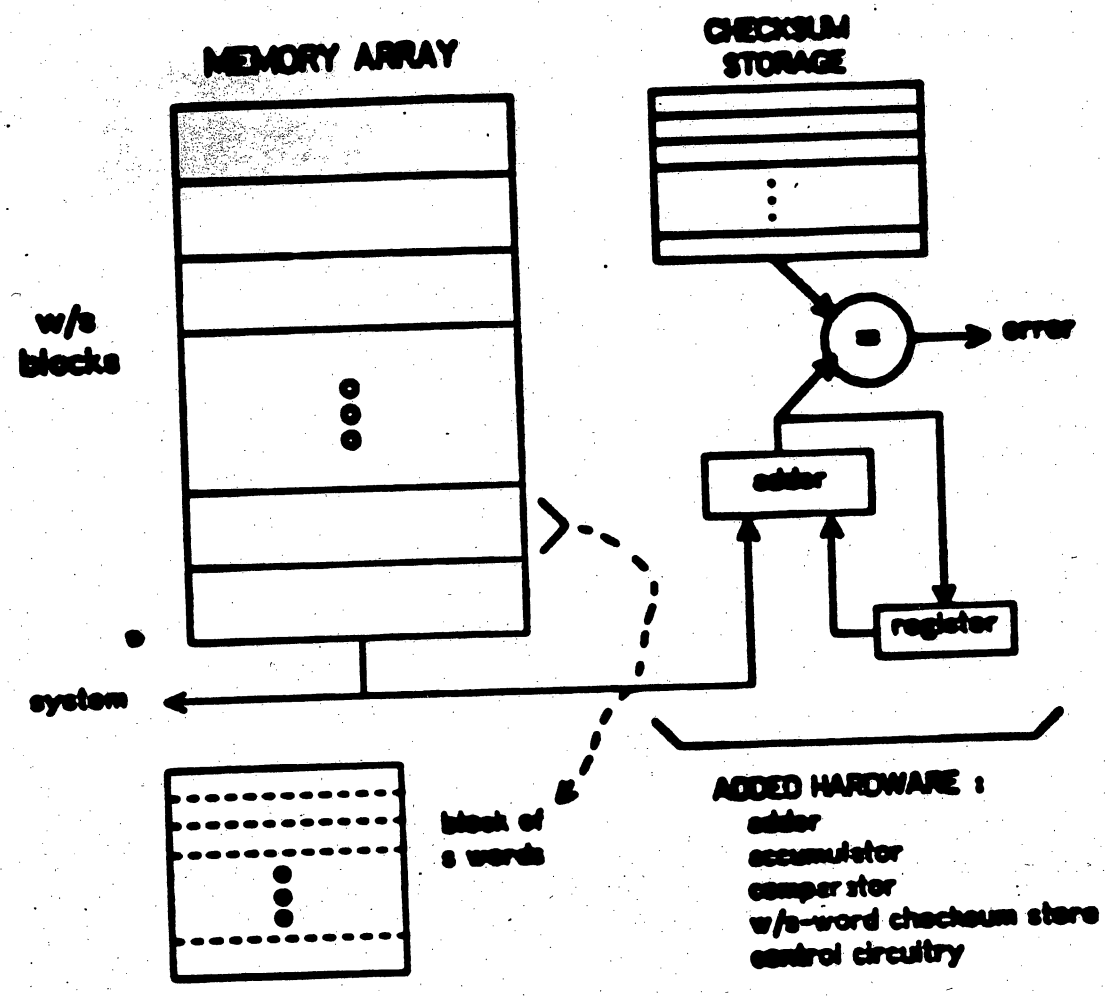


Figure 3.1 Memory with Checksum Error Detection

CHECK SUMS
(continued)

Honeywell checksum - modified double precision

adjacent words concatenated before addition

- overflow and loss of info

Low-cost residue codes

adder with end-around carry

Probability of Detection with Single Precision Checksum

Error Types	Coverage Ratio	Condition
Single Device - multiple word		
- Single column	1	$i < z - \log_2 s$
	$(1 - 2^{-s})$	$i = z - \log_2 s$
	$(1 - 2^{-s}(1 + c_{s/2}))$	
	$1 - 2^{-(z-i)} + 2^{-s})$	$i = z - \log_2 s + 1$ otherwise
- Multiple adjacent columns	1	$i \leq z - \log_2 s - c$
	$1 - 2^{-(z-i)} + 2^{-cs}$	otherwise
Multiple Device		
- Single word	1	
- Multiple word adjacent columns	$1 - 2^{-cs}$	$i \leq z - \log_2 s - c$
	$1 - [illegible]^{-(z-i)}$	otherwise

Probability of detection of errors through Honeywell Checksum

Error Types	Coverage Ratio	Condition
Single Word	1	
Multiple Word		
Single Column	1	$i < z - \log_2(s/2)$
	$(1-2^{-(s+1)})$	$i = z - \log_2(s/2)$
	$(1-2^{-s}(1+\frac{s}{s/2}))$	$i = z - \log_2(s/2) + 1$
	$(1-2^{-(z-i+(s/2))}+2^{-s})$	Otherwise
Multiple Column	1	$1 \leq z - \log_2(s/2) - c$
	$(1-2^{-(z-i+(\frac{s}{2^j}))}+2^{-cs})$	Otherwise

ARITHMETIC CODES

$C(A*B) = C(A) * C(B)$

invariance w.r.t. *

- usually ~~on~~ invariant w.r.t * ≠ not all operations needed (e.g. logic)

Simplest are AN codes:

$C(A) = aA$
where $a \neq 2^n$

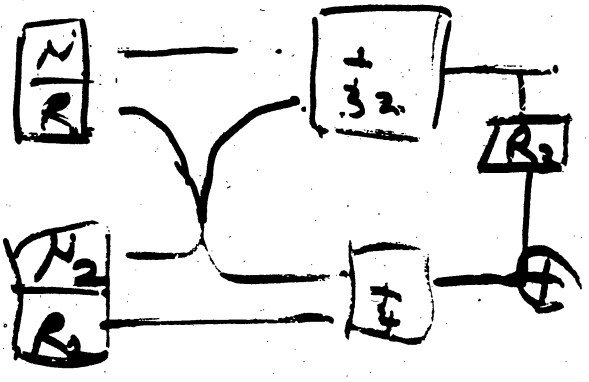
Simplest AN code:
$C(A) = 3N$

Residue Codes (~~Separable~~)

N	R(N)

Residue - M code word

$R(n) = N \bmod M$



Duplication

- Two copies, compare
  - disagreement indicates error.

    - cheap (comparison)
      twice $ + comparison
    - easy to implement
    - little perf. degr.
      - comparator
      - synch. delay

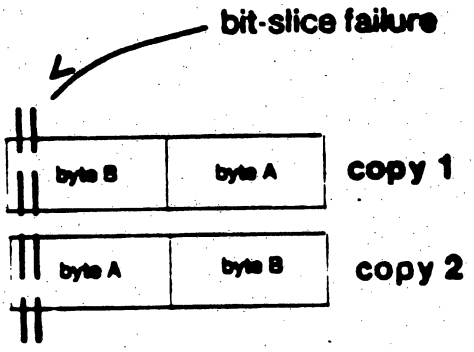
Coverage:
- All single failures (exc. comparator)
- Many multiple, as long as at least one non-overlapping

Variants on Duplication

1. Increase coverage
   - swap & compare - C.mmp (→)
   - dual-diversity
     - logical dual implementation (→)
     - "on-set" & "off-set" implementations
     - different algorithms

2. Application levels
   - Bus level (module) (Univac 1100/60 X→)
   - register (ESS-1)
   - processor (AXE)

3. Cost
   - Duplicate tasks, and do only part of processing in duplicate (TANDEM)
     coverage↓ cost ↓ perf ↑
   - Time redundancy
     coverage ↓↓ cost ↓ perf ↓



C.mmp, used for critical control data structures

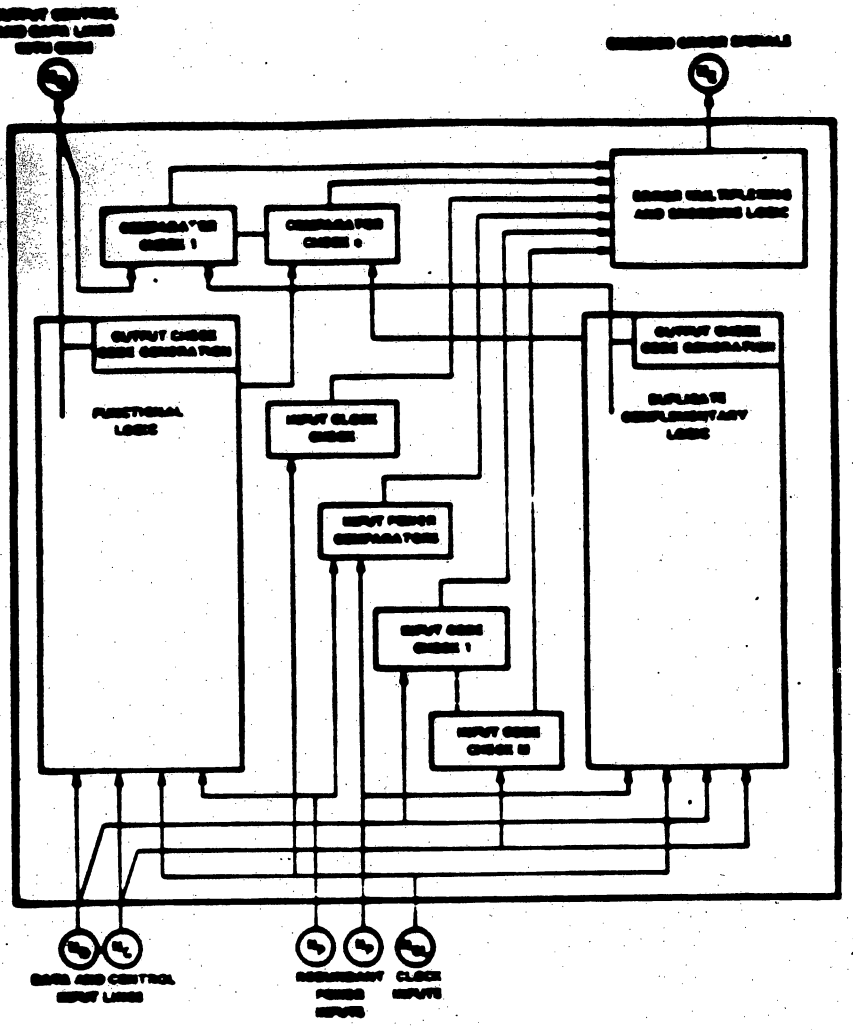


Fig. 1. Generalized VLSI chip.

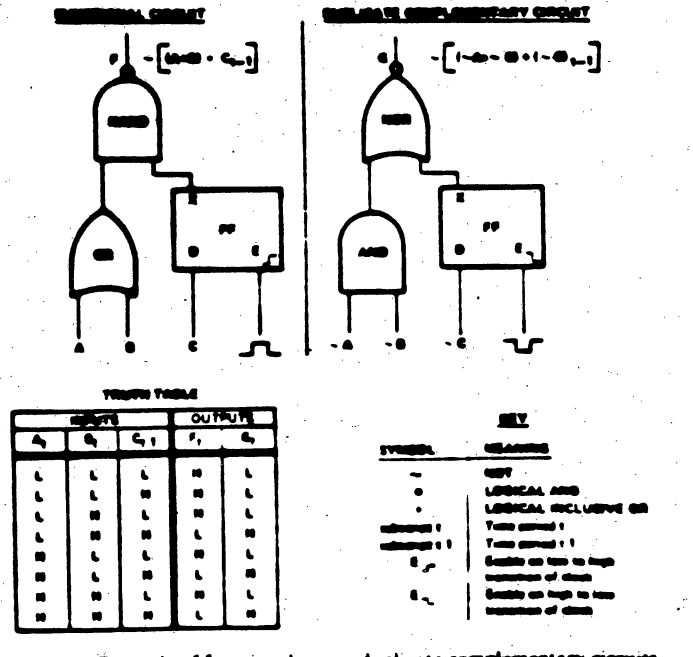


Fig. 2. Example of functional versus duplicate complementary circuits.

Error Correcting Codes

Extra bits not just used to detect, but also to overcome errors

- Triplication could be a code (an expensive one)

Codes -

terminology (n, k) codes

n = length of code word

k = length of actual data

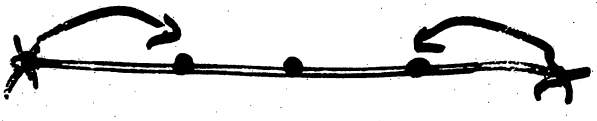
n-k = redundancy

Distance applies here as well

distance-d code can correct up to t errors $d \geq 2t+1$

or detect c errors, correct t' errors, where

$d \geq 2t'+c+1$



Hamming SEC (/DED codes)

↳ $d = 3$ ↳ $d = 4$

correct single bit errors

↓ code word vector

$$\begin{bmatrix} \text{decoding} \\ \text{matrix} \end{bmatrix} \begin{bmatrix} \quad \\ \quad \end{bmatrix} = \begin{bmatrix} \quad \end{bmatrix}$$

↑ Syndrome

	d_1	d_2	d_3	d_4	c_1	c_2	c_3	c_4
	1	1	1	1	1	1	1	1
	1	1	1	0	0	1	0	0
	1	1	0	1	0	0	1	0
	1	0	1	1	0	0	0	1
	0	1	1	1	0	0	0	1

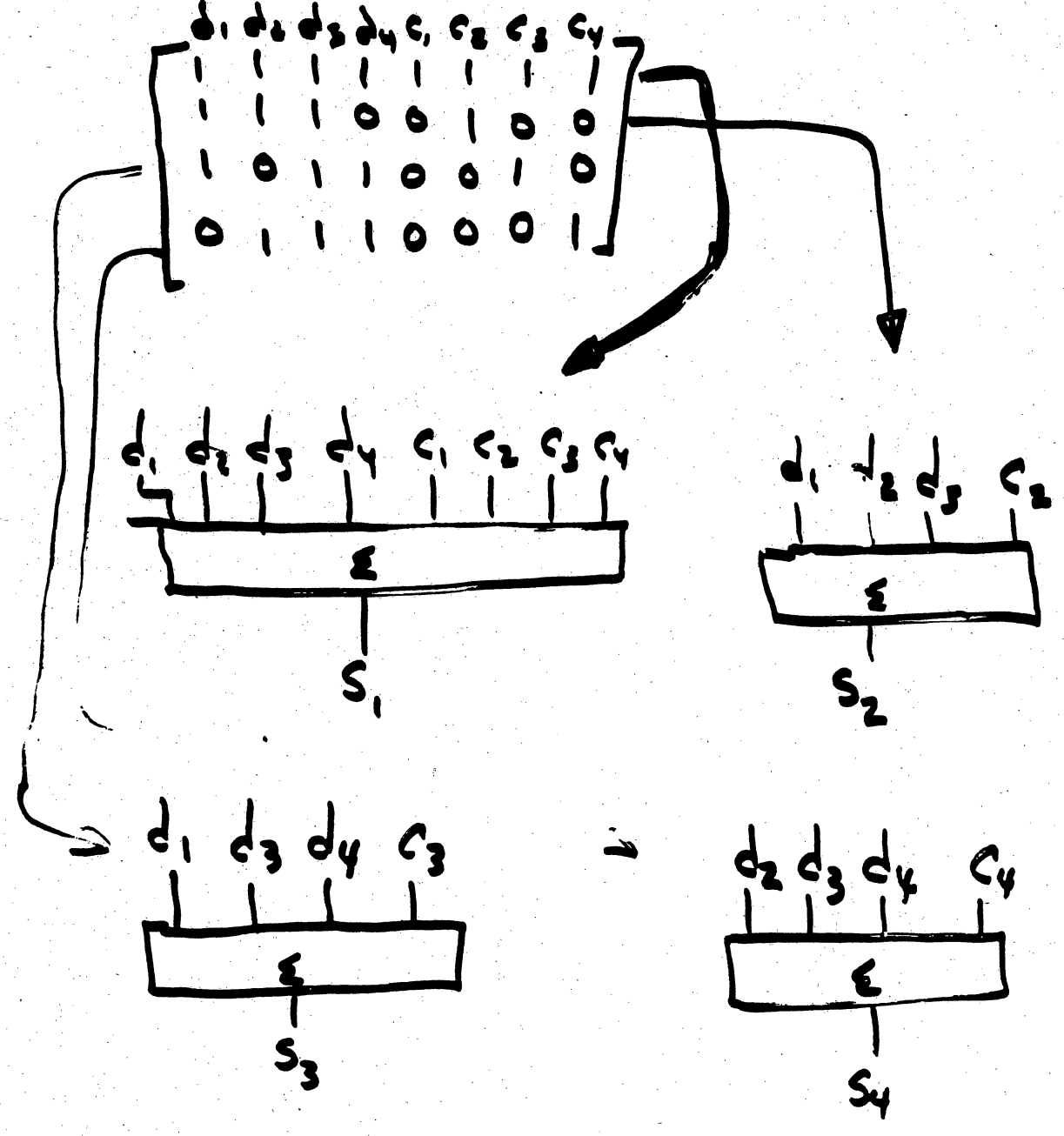
$\begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix} = [S_1\ S_2\ S_3\ S_4]$

(8,4)-code

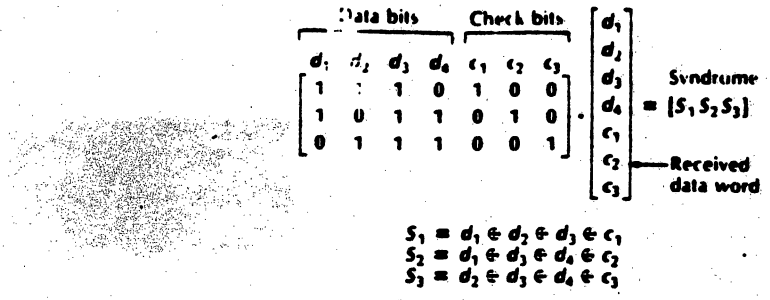
even parity

errors	received data bits	received check bits	syndrome
none	1110	0100	0000
one	1[0]10	0100	1101
two	1[0]10	0[1]00	0111

Decoding Matrix has a direct implementation in parity trees: one parity tree to produce each syndrome bit



Also: table lookup



a. Parity-check matrix and syndrome formation for a (7,4) Hamming SEC code.

	Data bits				Check bits			Syndrome	
Code word (no error)	1	1	1	0	1	0	0	0 0 0	(Zero syndrome implies no error)
One error (box)	1	1	1	[1]	1	0	0	0 1 1	(Matches d_4 column)
Two errors (boxes)	1	[0]	1	0	1	[1]	0	1 1 1	(Matches d_3 column—results in erroneous correction)

b. Received code words and their syndromes for zero, one, and two errors.

c_1	c_2	d_1	c_3	d_2	d_3	d_4
1	0	1	0	1	0	1
0	1	1	0	0	1	1
0	0	0	1	1	1	1

c. Parity-check matrix for (7,4) Hamming code for which syndrome is the binary-coded position of the bit in error.

Figure 3–57. Hamming SEC code examples.

$$\begin{array}{c} \begin{matrix} d_1 & d_2 & d_3 & d_4 & c_1 & c_2 & c_3 & c_4 \end{matrix} \\ \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array} \cdot \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix} = [S_1 S_2 S_3 S_4]$$

a. Parity-check matrix for (8,4) Hamming SEC/DED code.

Number of errors	Received data bits				Received check bits				Syndrome			
	d_1	d_2	d_3	d_4	c_1	c_2	c_3	c_4	S_1	S_2	S_3	S_4
Zero	1	1	1	0	0	1	0	0	0	0	0	0
One	1	[0]	1	0	0	1	0	0	1	1	0	1
Two	1	[0]	1	0	0	1	[1]	0	0	1	1	1

b. Received words and their syndromes.

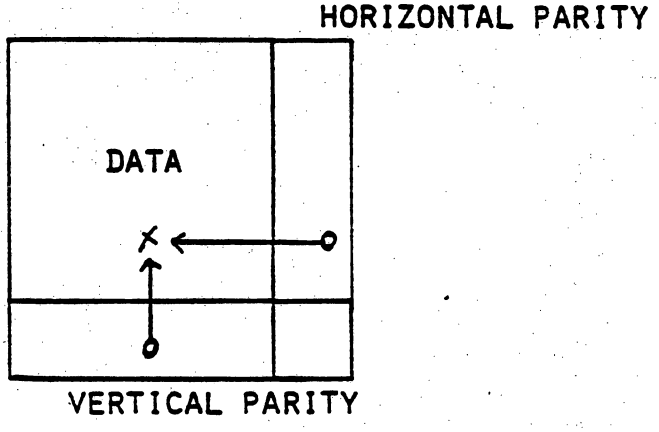
Figure 3–58. Hamming SEC/DED code examples.

ON-LINE CORRECTION

HAMMING CODE

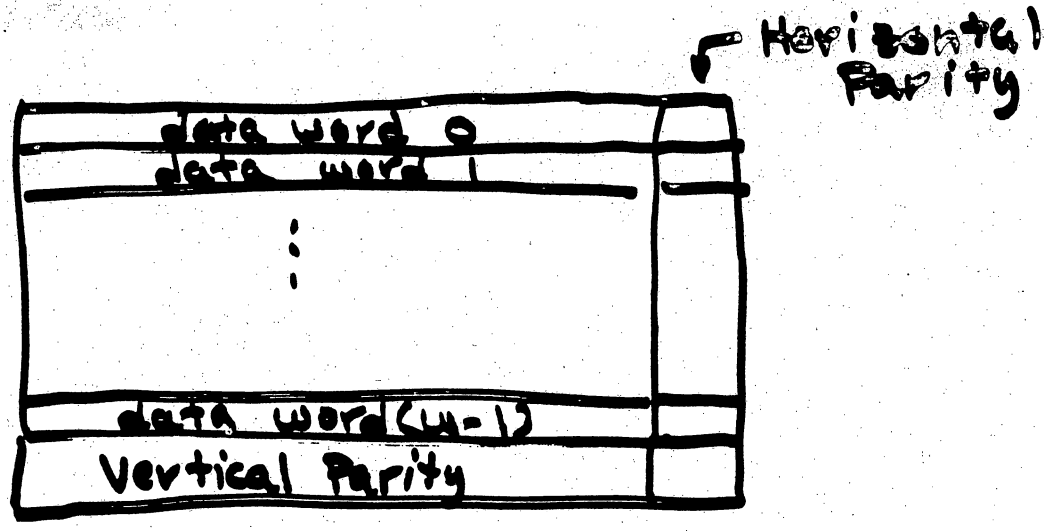
- SEC (BIT)
- SEC/DED (BIT)
- SEC (BIT)/DED (BYTE)
- SEC (BYTE)
- SEC (BYTE)/DED (BYTE)

BLOCK CODE CORRECTION



TRIPLICATION

Block code



special case of a product code

weight (product code)

$$= \prod_{i=1}^{n} \text{weight (code } i\text{)}$$

$d \geq 2t+1$

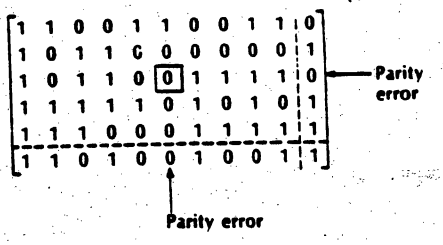


Figure 3–62. Product code using two even-parity codes.

Some other ECC codes

Whole range of cyclic codes
- multiple error
- burst error
- byte error

Modified Cyclic codes

The greater the distance, the more complex (expensive, time consuming) the decoding

⇒ more than d=4 extremely rare for ECC codes

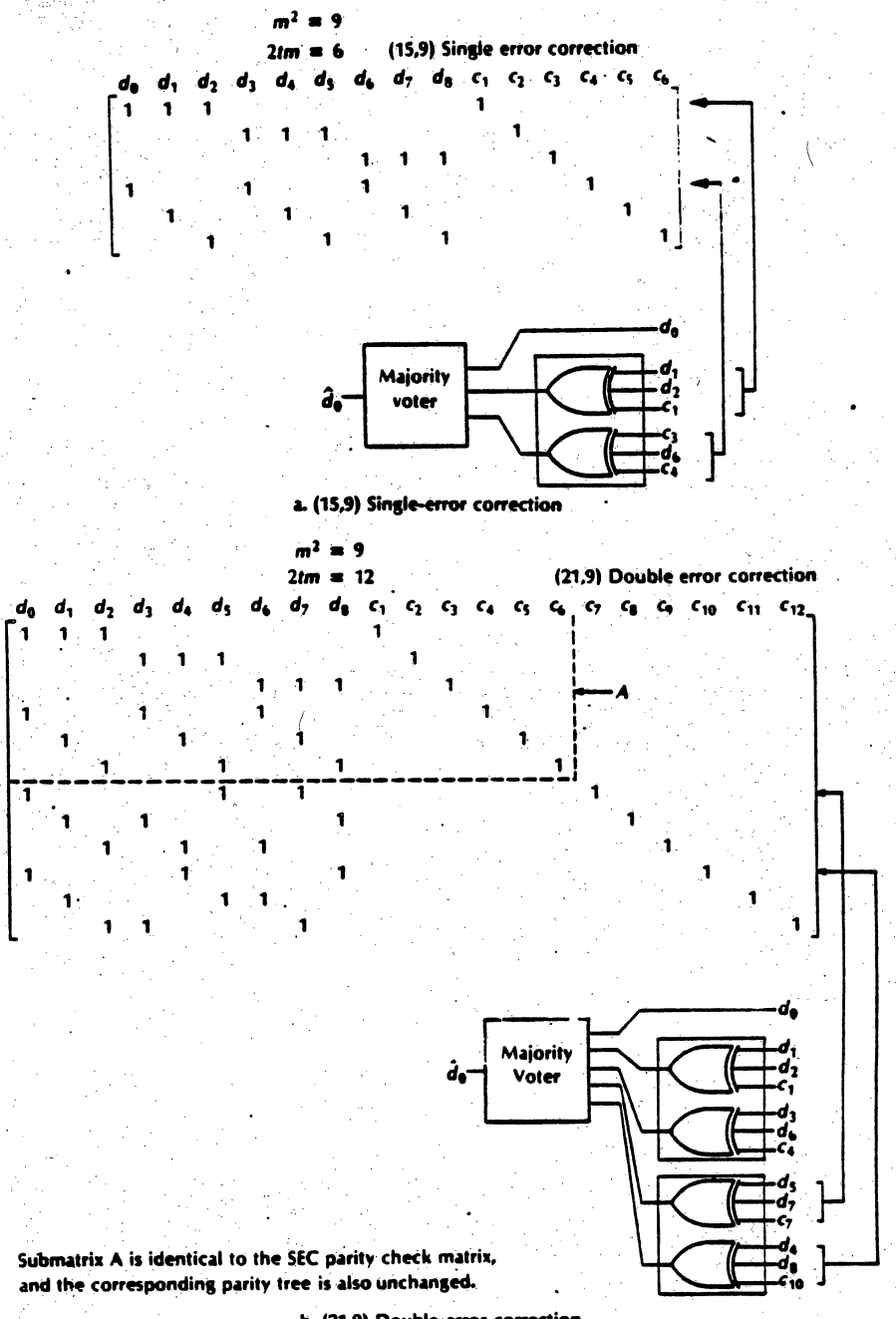


Figure 3–60. Latin square code parity-check matrix with one bit-slice of decoder for nine data bits. Decoding is performed by a majority vote among the received value of a data bit and two values calculated for it from the other received bits. a.) Single-error correction. b.) Double-error correction.

	Bit planes				
Address	0	1	2	3	Bit location on plane mapped into address SEC code memory
0	0	0	0	0	Initial configuration: two single-bit failures (boxes) and a third failure (circle) cause a double (noncorrectable) error
1	1	1	(1)	[1]	
2	2	2	2	2	
3	3	3	3	[3]	
0	0	(1)	2	[3]	Second configuration: with three tolerable single-bit failures, a fourth, noncorrectable failure occurs (circle).
1	1	0	3	2	
2	2	3	0	[1]	
3	3	2	[1]	0	
0	0	2	3	[1]	Third configuration: the fourth failure is no longer aligned with any other failure; however, another configuration is needed because two old failures are aligned.
1	1	3	2	0	
2	2	0	[1]	[3]	
3	3	(1)	0	2	
0	0	3	[1]	2	Fourth configuration: no double failures exist, but any additional failure is unrecoverable.
1	1	2	0	[3]	
2	2	[1]	3	0	
3	3	0	2	[1]	

Figure 3–84. Orthogonal Latin squares–based memory address skewing used to reconfigure a bit-sliced SEC code memory.

Table 3–13. Single-error-correcting AN codes.

Check Modulus A	*Maximum Data Length k*	*Code Word Length n*
13	2	6
19	4	9
23	6	11
29	9	14
37	12	13
47	17	23
53	20	26
59	23	29
61	24	30
67	26	33
71	28	35
79	32	39
83	34	41
101	42	50
103	43	51

Source: Kautz [1962]

Reconfiguration

- Based upon error detection
- coverage important factor in modelling
- rollback strategies -
  process recovery after reconfiguration
  - state checkpointing

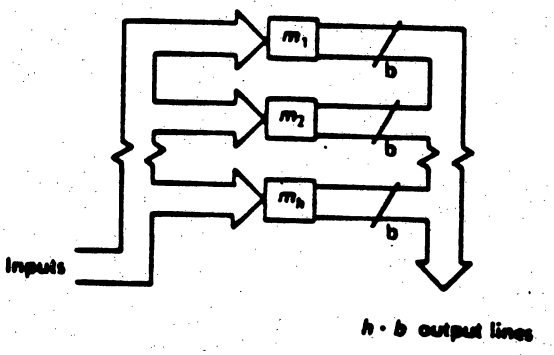
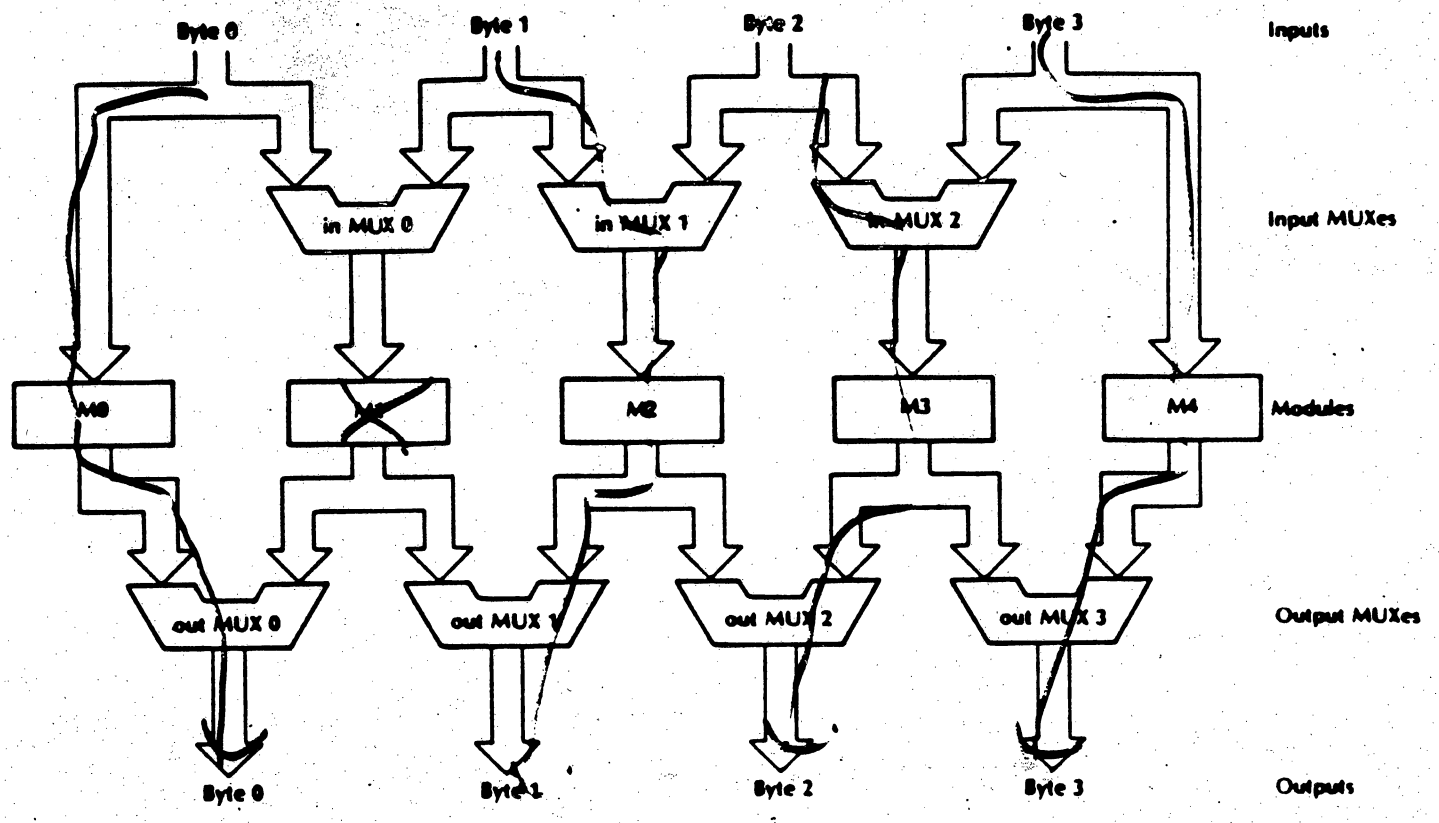


Figure 3–59. Circuit design of parallel byte-wide modules.



Control input to MUX: 0 (left leg), 1 (right leg)

Normal control line state: (in MUX 0, in MUX 1, in MUX 2) = (1, 1, 1)
(out MUX 0, out MUX 1, out MUX 2, out MUX 3) = (0, 0, 0, 0)

Control line state if M2 failed: in MUX = (1,X,0)
out MUX = (0,0,1,1)
X = don't care

Figure 3-19. Possible implementation of a system made from four byte slice ... added as a spare.

GUARANTEED

Evaluation Criteria

1. What are some good measures of "reliability"?

2. How can we model systems to evaluate their potential "reliability"?

Deterministic Model

System must tolerate k failures. (k = figure of merit)

=> potentially wasteful allocation of resources--some components have much higher failure rates than others

Probabilistic Models

- based on component failure distributions
- several levels of detail
  - chip (gate, component) level
  - module level (often nonredundant)
  - system level

Redundant System Models

The form of the model depends upon the assumptions made:

- failure to exhaustion of spares

  => combinatorial model

- failure with repair

  => Markov model

- failure with periodic maintenance

  => Markov model

Evaluation Criteria

Probabilistic Functions

- Reliability $R(t)$
- Mission Time $MT(r)$

Single Parameters (Probabilistic)

- Mean Time To Failure (MTTF)

Comparative Measures (Probabilistic)

- Reliability Difference $R_2(t) - R_1(t)$
- Unreliability Gain $\frac{1 - R_2(t)}{1 - R_1(t)}$
- Mission Time Improvement $\frac{MT_2(r)}{MT_1(r)}$

Other Measures

Mean Time To Failure

$$MTTF = \int_0^\infty R(t)dt$$

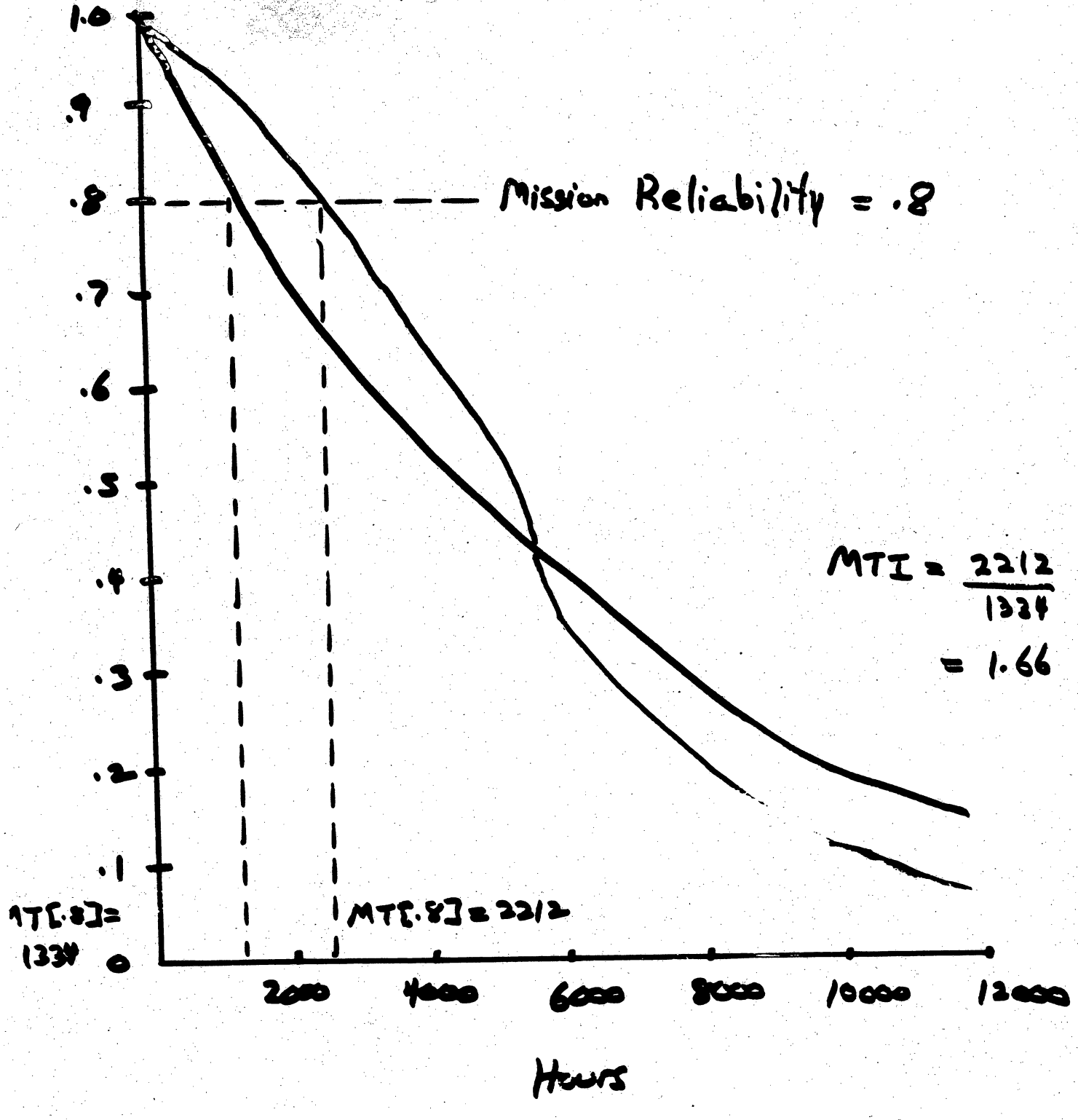
Mission Time MT(r)

$$\text{Exponential: } MT(r) = \frac{\ln(r)}{\lambda}$$

Mission Time Improvement

$$MTI = \frac{MT_2(r)}{MT_1(r)}$$

- Repairability
- Diagnosibility
- Testability
- Mean Time To Repair (MTTR)


1.0
.9
.8
.7
.6
.5
.4
.3
.2
.1
0
Mission Reliability = .8
MTI = 2212 / 1334
= 1.66
MT[.8] = 1334
MT[.8] = 2212
2000
4000
6000
8000
10000
12000
Hours

Other Measures cont.

Availability A(t)

steady state $\frac{MTTF}{MTTF + MTTR}$

Coverage

1. Extent of error recovery provided
2. Conditional probability of recovery given that an error occurs

Reliability Difference

$$R_2 - R_1$$

Reliability Gain

$$\frac{R_2}{R_1}$$

Reliability Improvement Factor

$$\frac{1 - R_1}{1 - R_2}$$

Improvement Index

$$\frac{\log R_2}{\log R_1}$$

```
module [16k.ecc.mem:
        1, [ecc.control:
                6.SSIMISC(..@ic_theta (pkg, air_flow));
                6.MSIMISC(..@ic_theta (pkg, air_flow))
            ];
        1, [ecc.bus.conn: 12F.DEC8881(..@ic_theta (pkg, air_flow)) ];
        1, [extra.support:
                10.MSIMISC(..@ic_theta (pkg, air_flow));
                18F.74174(..@ic_theta (pkg, air_flow));
                1.ECC18(..@ic_theta (pkg, air_flow));
                18F.7486(..@ic_theta (pkg, air_flow));
                3.7404(..@ic_theta (pkg, air_flow));
                1.74154(..@ic_theta (pkg, air_flow));
                18F.74157(..@ic_theta (pkg, air_flow))
            ];
        1, [ecc.array:  rel.fn          !  18-bit ECC, 4 rows of 18 chips
                equation =
                    < @sec_mem (18, 16384, f_mode, R[array_chip], 4096, 256)>;
                partlist = < 72, $array_chip >
            ];
       ];

module [array_chip: 1, mk4096(..@ic_theta (pkg, air_flow))];


!now try adding ecc memory (ignore failure process of array)

replace 16k.memory with 16k.ecc.mem;
let f_mode = 1;      ! whole chip failure mode


ttyprint ("doing mt with ecc memory (B1)");
mtime [pdp8_e] at 0.95;
let mt_b1 = sys;
subtitle SEC memory: design B1;
lambda[pdp8_e];
ttyprint ("doing mttf analysis of version B1");
mttfanal [pdp8_e];
let mttf_b1 = sys;
```

PDP-8 redesign analysis
SEC memory: design B1

n Quantity) Module		Lambda - single Module	% Lambda - all module copies
1) PDP8_E	R	14.6479	100.000
1) CPU		9.7628	66.650
1) DATA_PART		4.6454	47.583
1) REGISTERS		1.8614	40.069
1) A.MQ.MB.PC.MA		1.8374	98.713
1) LINKBIT		0.0240	1.287
1) DATA.PATH		2.7840	59.931
1) ADDER		0.6751	24.250
1) T.C.1.0		0.1291	4.638
1) PATH.SHUNT		1.9798	71.112
1) CPU.BUSCONN		0.1794	1.838
1) MB300.OC		0.1345	75.000
1) MB310.OC		0.0448	25.000
1) BUS.LDS.MB320		0.0238	0.244
1) CONTROL.LOGIC		4.9142	50.336
1) MB300.CTL		0.1224	2.491
1) MB310.CTL		1.4423	29.350
1) TIM_GEN		3.3495	68.159
1) MEMEXT.TIMSHR		2.1555	14.716
1) 16K.ECC.MEM	R	2.7296	18.634
1) ECC.CONTROL		0.4639	16.994
1) ECC.BUS.CONN		0.0673	2.465
1) EXTRA.SUPPORT		2.1984	80.541
1) ECC.ARRAY	RF	0.0000	0.000
72) ARRAY_CHIP		1.0691	N/A

Flags after module name:
"*" - possible default parameter change
"R" - module has redundant submodules
"RF" - module is defined by REL.FN

**** Nonredundant portion MTTF: 68269.3 hrs. ****
**** Mean Time to Part Failure: 10914.3 hrs. ****
Est. number parts: 336.1 Est. number pins: 5146

**** WARNING - rollup failure rate-based statistics which follow may have
**** no usefulness since the system is redundant

Est. Costs: Net Part Cost: 0.0000
Repair Cost Rate : 0.0000 per million Hrs.
Mean Material Repair Cost: 0.0000
with 38 zero-cost part numbers

Sortkey summary statistics :

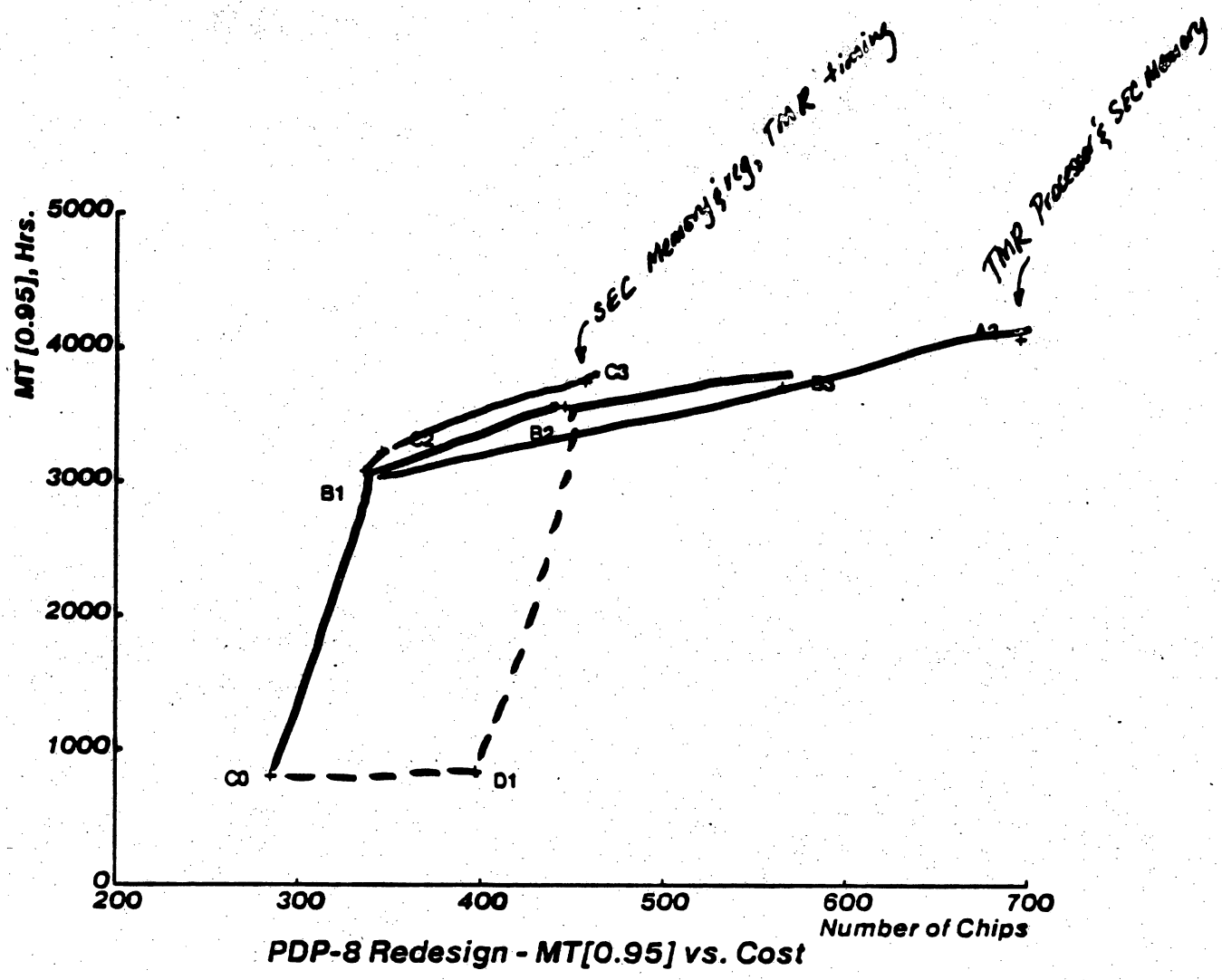
PDP-8 redesign analysis
SEC memory: design B1

MTTF Analysis of Module: PDP8_E

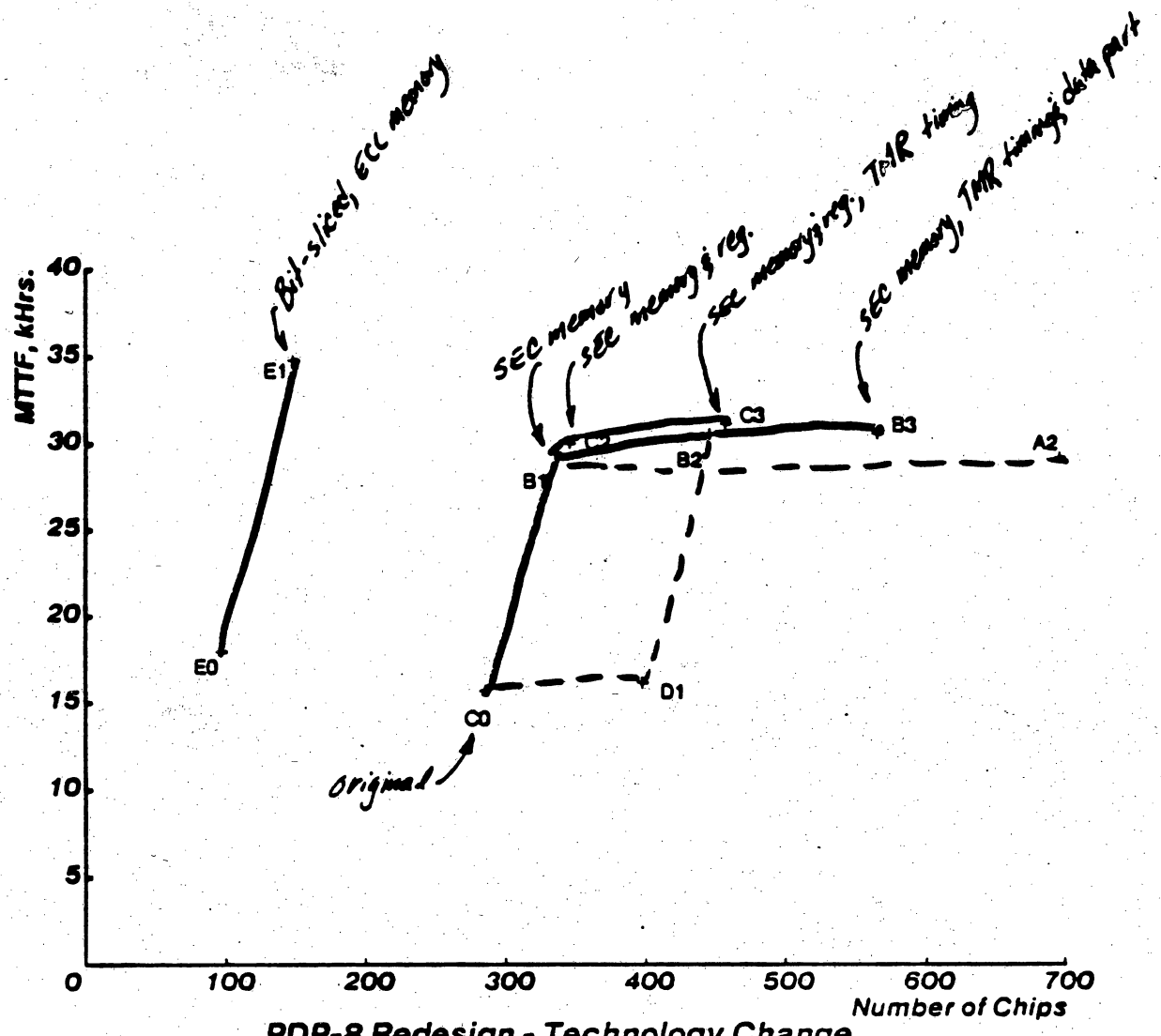
Quantity) Module		MTTF for Single Module	
1) PDP8_E	R	2.936E+04	29359.4
1) CPU		1.024E+05	102442.6
1) DATA_PART		2.153E+05	215291.9
1) REGISTERS		5.373E+05	537296.4
1) A.MQ.MB.PC.MA		5.443E+05	544299.3
1) LINKBIT		4.175E+07	41754209.6
1) DATA.PATH		3.592E+05	359238.3
1) ADDER		1.481E+06	1481377.5
1) T.C.1.0		7.746E+06	7745870.8
1) PATH.SHUNT		5.052E+05	505176.8
1) CPU.BUSCONN		5.575E+06	5575114.0
1) MB300.OC		7.433E+06	7433307.1
1) MB310.OC		2.230E+07	22299906.1
1) BUS.LDS.MB320		4.202E+07	42022105.7
1) CONTROL.LOGIC		2.035E+05	203520.3
1) MB300.CTL		8.169E+06	8168939.2
1) MB310.CTL		6.934E+05	693416.1
1) TIM_GEN		2.986E+05	298594.8
1) MEMEXT.TIMSHR		4.640E+05	463977.5
1) 16K.ECC.MEM	R	3.974E+04	39742.9
1) ECC.CONTROL		2.156E+06	2156086.7
1) ECC.BUS.CONN		1.487E+07	14866692.7
1) EXTRA.SUPPORT		4.549E+05	454928.1
1) ECC.ARRAY	RF	4.304E+04	43036.0
72) ARRAY_CHIP		9.355E+05	935483.1

Flags after module name:
"*" - possible default parameter change
"R" - module has redundant submodules
"RF" - module is defined by REL.FN

**** MTTF: 29359.4 hrs. ****
**** Mean Time to Part Failure: 10914.3 hrs. ****
Est. number parts: 336.1 Est. number pins: 5146



PDP-8 Redesign - MT[0.95] vs. Cost



PDP-8 Redesign - Technology Change

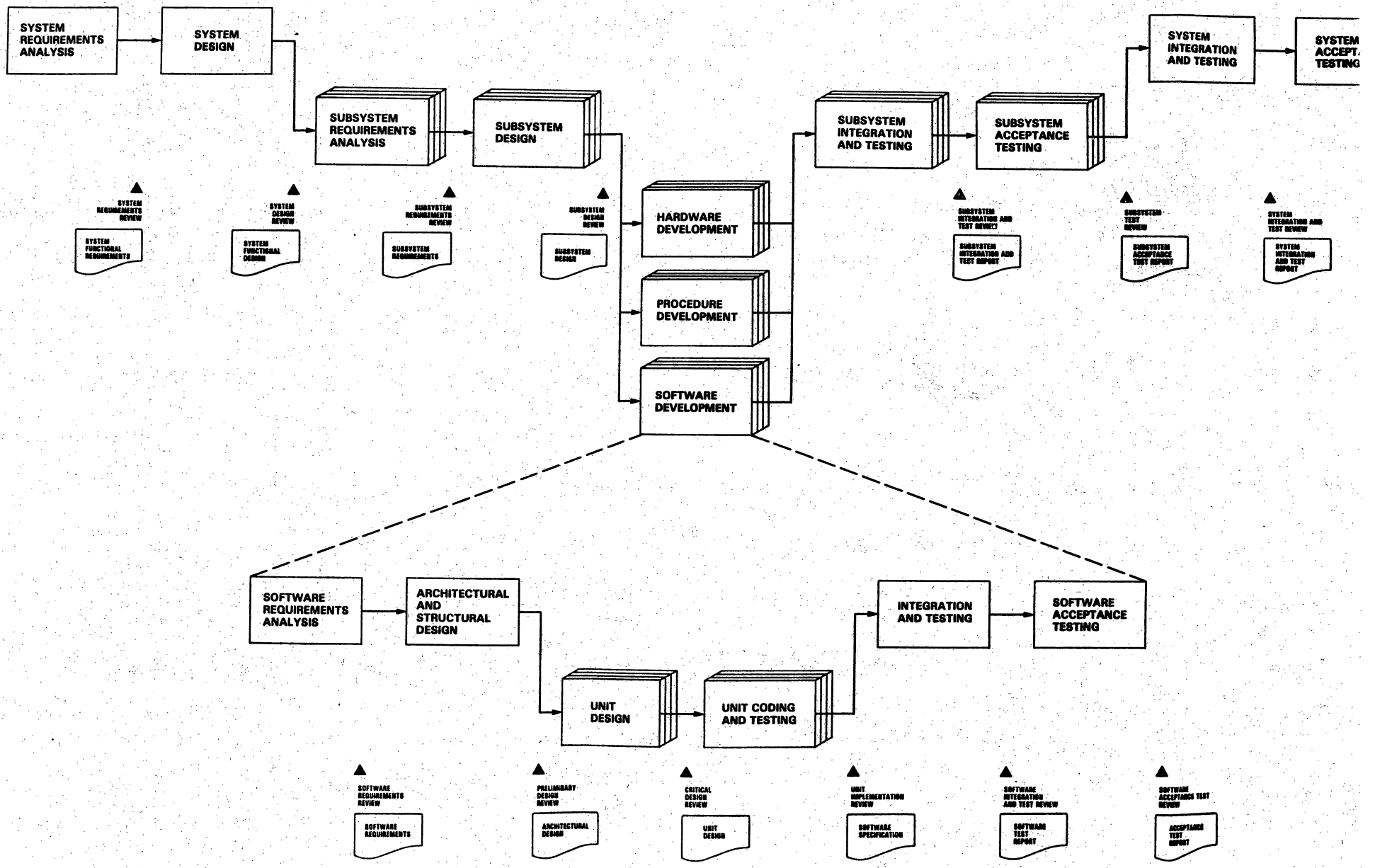
Reliability Evaluation:
Reliability Measures

- Reliability function (R(t))
- Hazard function (z(t))
- "Mean time" measures (MTTF, MTBF, MTTC, MTBC)
- Mission measures (MT(r), MR(t), MTI(r))
- Repairability (MTTR, diagnosability)
- Availability (A(t), A)
- Coverage (fault detection and recovery)
- Cost (LCC, BMC)
- Performance (MCBF, $E[t_{complete}]$, performability)

4.1) System Specification

JPL

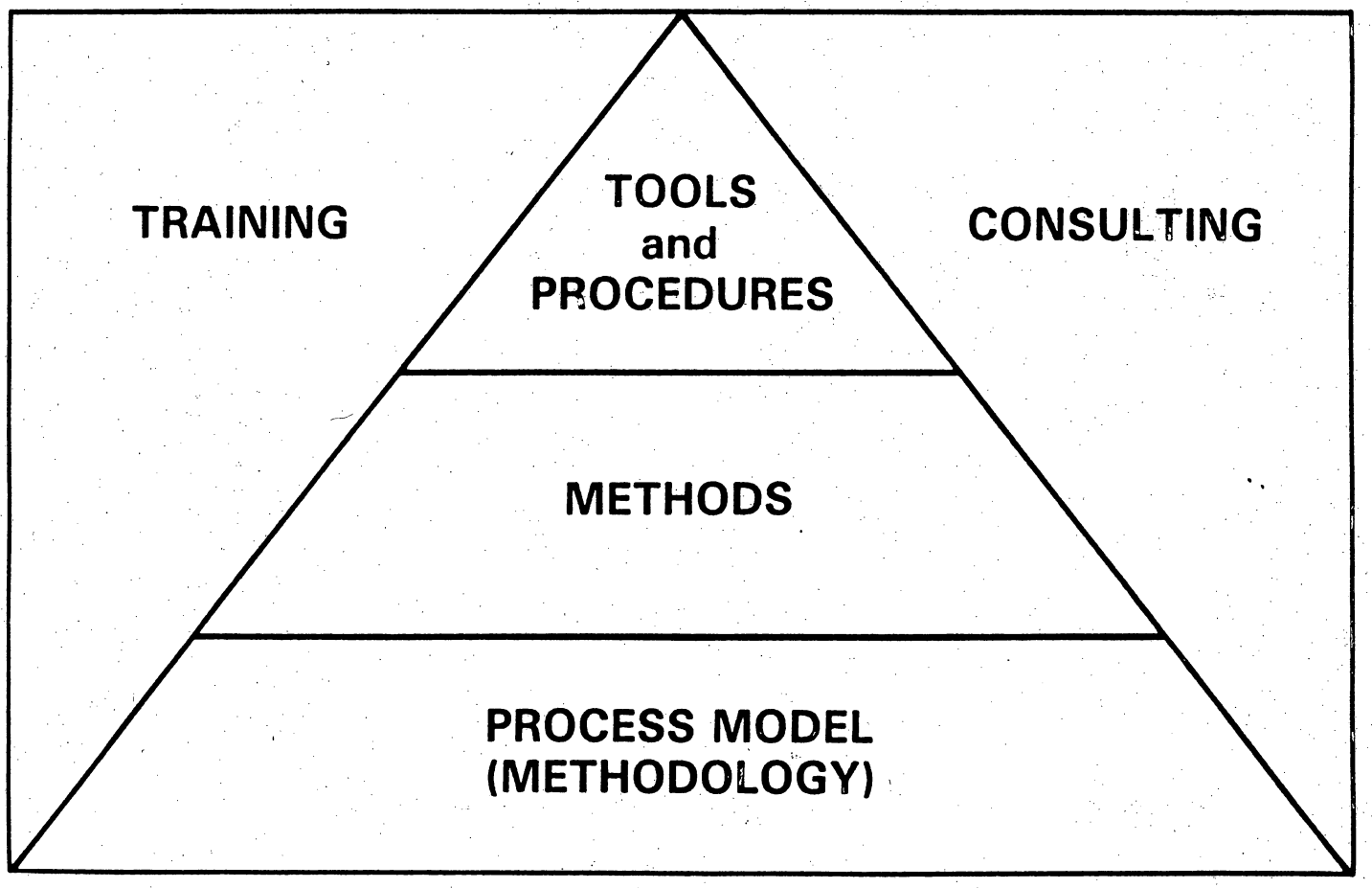
INFORMATION SYSTEM LIFE CYCLE



JPL Software Intensive System Methodology

INFRASTRUCTURE FOR SOFTWARE INTENSIVE SYSTEM DEVELOPMENT

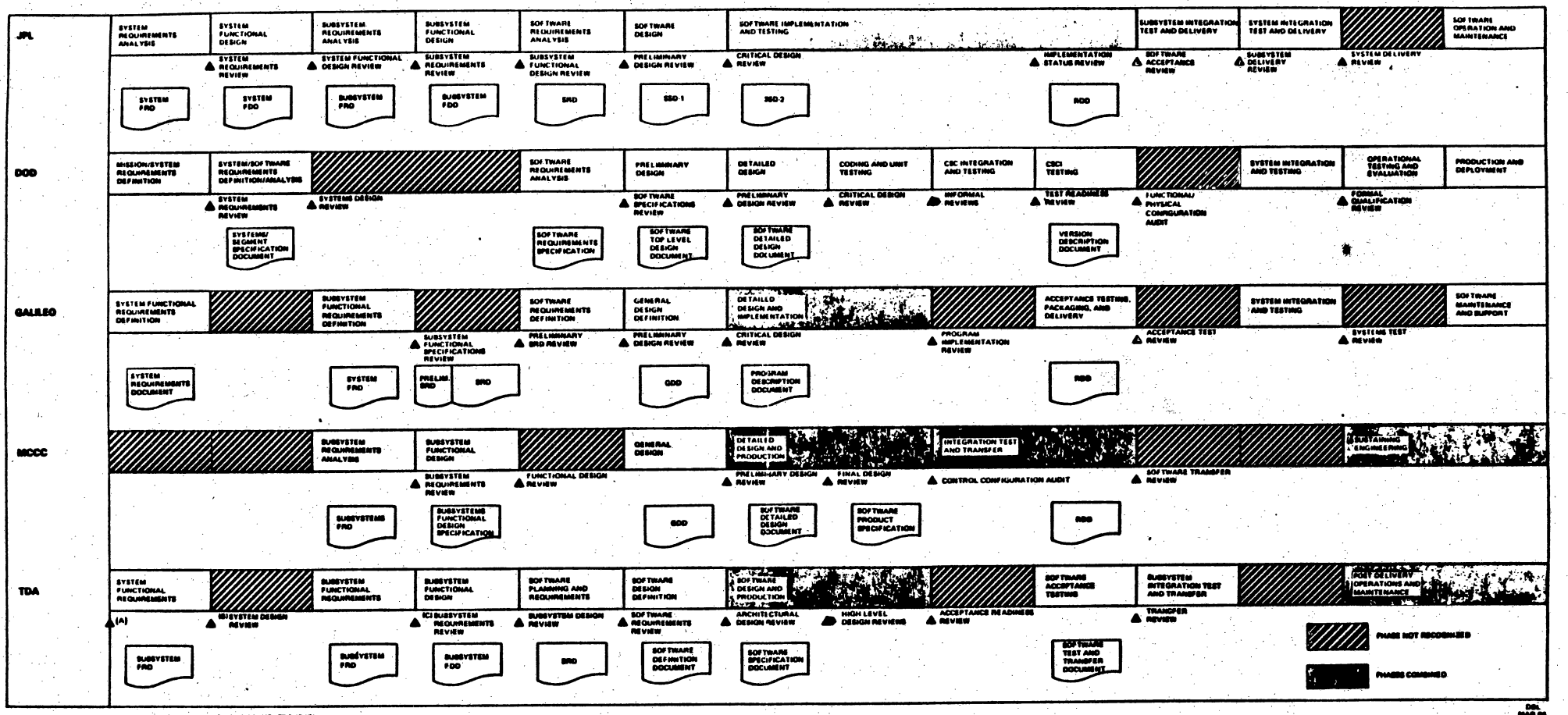
JPL



V0025
Y0008

DBL
MAR 86

LIFE CYCLE COMPARISON



ONLY REQUIREMENTS AND DESIGN DOCUMENTS SHOWN

FRD = FUNCTIONAL REQUIREMENTS DOCUMENT
FDD = FUNCTIONAL DESIGN DOCUMENT
SRD = SOFTWARE REQUIREMENTS DOCUMENT
SSD = SOFTWARE SPECIFICATION DOCUMENT
GDD = GENERAL DESIGN DOCUMENT
RDD = RELEASE DESCRIPTION DOCUMENT

DSN
SMAP 05

comet.REL LSI- 16.000 ROM- 16.000 RAM- 16.000

E - 1.000 D - 16.000 L - 1.000 T - 40.000

MODULE	FAILURE RATE	PERCENTAGE
COMET.CPU.PLUS.MEMORY.CONTROL	225.755	100.000
Data.Path.Module	36.543	16.187
Misc	15.583	42.643
ROM.AND.RAM	9.890	27.087
GATE.ARRAY	11.062	30.271
Memory.Interface.Cache	56.525	25.038
Misc	16.573	30.003
ROM.AND.RAM	27.490	48.648
GATE.ARRAY	12.067	21.349
Unibus.Interface	31.945	14.150
Misc	18.731	58.633
ROM.AND.RAM	6.678	20.905
GATE.ARRAY	6.537	20.462
Control.Store	67.518	29.908
Misc	3.828	5.667
ROM.AND.RAM	63.698	94.343
Memory.Controller	33.224	14.717
Misc	21.319	64.169
ROM.AND.RAM	8.385	25.237
GATE.ARRAY	3.520	10.594

# of chips - 662.000 # of gates - 33361.000 # of bits - 732416.000

- -

SUMMARY ROLLUP BY COMPONENT TYPE

TYPE	# of CHIPS	FAILURE RATES	PERCENTAGE
SSI	180.000	24.436	10.824
MSI	142.000	51.492	22.809
LSI	67.000	33.669	14.914
ROM	158.000	79.419	35.179
RAM	123.000	36.739	16.274
MOS	.000	.000	.000
BIP	662.000	225.755	100.000

1 may 1979

combcc.rel[x330ds73] LSI- 16.000 ROM- 16.000 RAM- 16.000

E - 1.000 Q - 16.000 L - 1.000 T - 40.000

MODULE	FAILURE RATE	PERCENTAGE
COMET.CPU.PLUS.MEMORY.CONTROL	163.566	100.000
Data.Path.Module	36.543	22.342
Misc	15.583	42.643
ROM.AND.RAM	9.898	27.087
GATE.ARRAY	11.062	30.271
Memory.Interface.Cache	56.525	34.558
Misc	16.959	30.003
ROM.AND.RAM	27.498	48.648
GATE.ARRAY	12.067	21.349
Unibus.Interface	31.945	19.531
Misc	18.731	58.633
ROM.AND.RAM	6.678	20.905
GATE.ARRAY	6.537	20.462
Control.Store	5.328	3.257
Misc	3.820	71.689
GATE.ARRAY	1.508	28.311
Memory.Controller	33.224	20.312
Misc	21.319	64.169
ROM.AND.RAM	8.385	25.237
GATE.ARRAY	3.520	10.594

# of chips = 545.000 # of gates = 34561.000 # of bits = 240896.000

- -

SUMMARY ROLLUP BY COMPONENT TYPE

TYPE	# of CHIPS	FAILURE RATES	PERCENTAGE
SSI	180.000	24.436	14.940
MSI	142.000	51.492	31.481
LSI	70.000	35.177	21.507
ROM	30.000	15.721	9.611
RAM	123.000	36.739	22.461
MOS	.000	.000	.000
BIP	545.000	163.566	100.000

BITS PER WORD	WORDS IN MEM.	MODE	BITS PER CHIP	ROWS PER CHIP	CHIPS IN MEM.	FAILURE RATE CHIP(N)	FAILURE RATE CHIP(I)	FAILURE RATE CONTROL	FAILS PRE-SENT	MTTF
1377	384	2	4096	-	129	0.5300	0.0000	0.0000	0	368913.2
1377	384	2	4096	-	129	0.5300	0.0000	3.8000	0	181031.5
2673	192	2	4096	-	125	0.5300	0.0000	0.0000	0	271714.8
2673	192	2	4096	-	125	0.5300	0.0000	3.8000	0	155771.6

Results of Initial VAX-11/750 study

- Temperature sensitivity help justify heat sink
- For a 2 Mbyte system main memory accounts for 71% of system failure rate. Hamming code improved CPU/Memory MTTF by a factor of almost 3.5.
- Control store represented 57% of CPU failure rate of which 82% was microstore for 47% of CPU.
- 56% of 3 board CPU failure rate in ROM/RAM.

Gate Array Junction Temperature	Relative Failure Rate of CPU
50°C	1.00
60°C	1.05
70°C	1.15
80°C	1.30

ITERATIONS OF THE DESIGN CYCLE

	% λ	λ PER 10^6	MTTF HRS
STOCK COMET	-	225	4444
ECC CONTROL STORE	28	163	6135
ECC MEMORY INTERFACE/CACHE	12	130	7692
ECC DATA PATH	4	111	9009
ECC UNIBUS INTERFACE	3		
ECC MEMORY CONTROLLER	3		

Some Examples of VAX-11/750 RAMP

- Fault intolerant techniques such as single cabinet, bounded system
- Control store 2 parity bits (one even, one odd) to isolate failures to 10 chips.
- Graceful degradation through parity on cache (5 parity bits) and disable; parity on translation buffer halves (8 parity bits) and disable.

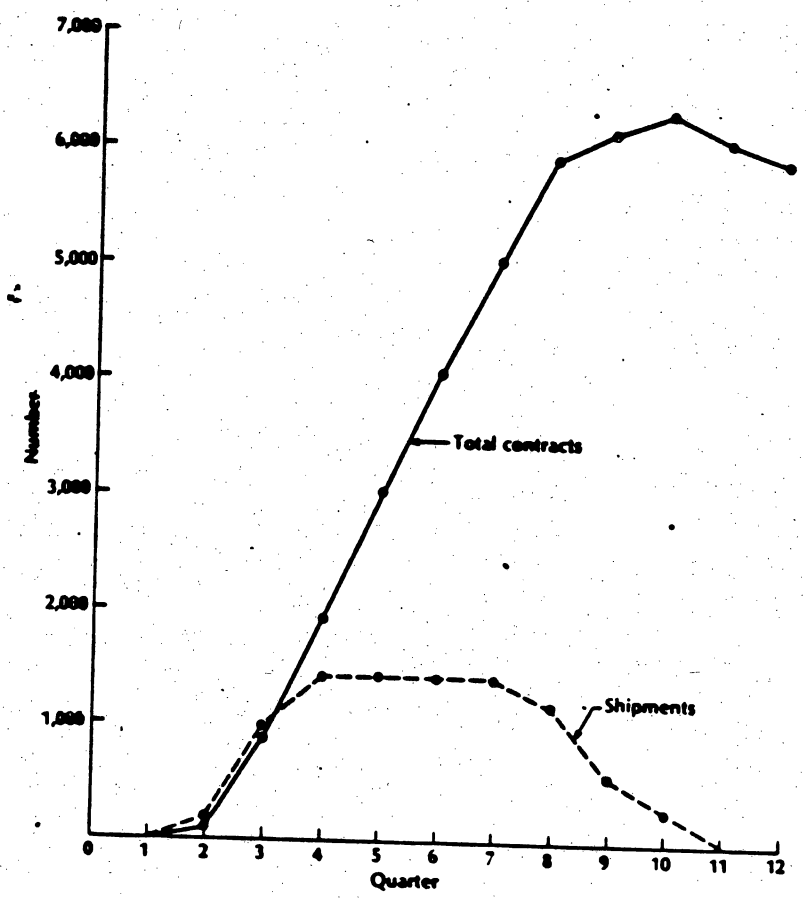


Figure 6–1. Example of installed base curve.

Table 6–1. Example of installed base calculation.

Quarter	*Shipments*	*New Contracts*	*Contract Attrition*	*Total Contracts*
1	60	45	0	45
2	200	150	0	195
3	1,000	750	0	945
4	1,400	1,050	0	1,995
5	1,400	1,050	5	3,040
6	1,400	1,050	15	4,075
7	1,400	1,050	75	5,050
8	1,200	900	105	5,845
9	600	450	109	6,186
10	300	225	119	6,292
11	0	0	173	6,119
12	0	0	185	5,934

75% contract penetration
90% renewal rate

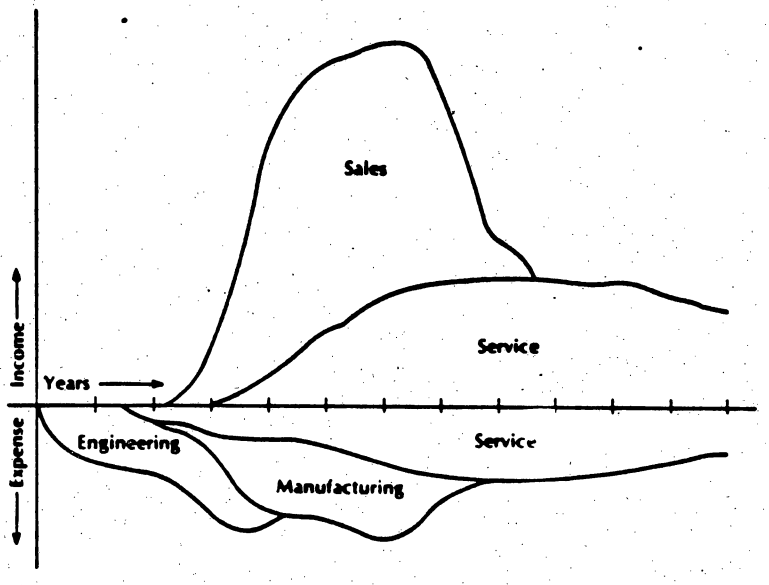


Figure 6–4. Cost and income distribution over the life of a system.

Sources of Maintenance Costs

- Labor Expense

  $ALE = CPH(8760)[(MTTR + TTR)/MTBF + (MTPM + TTPM)/MTBP]$

- Material Expense
- Inventory Costs
- Level of Service
- Training Costs
- Depreciation of Capital Equipment

Table 6–2. Example of cost of ownership calculation (in thousands of dollars).

(1) Year	(2) Maintenance Cost	(3) Depreciation	(4) Net	(5) After Tax Cash Flow	(6) Present Value Factor	(7) Discounted Cash Flow
1	100	400	−300	−150	0.909	−136.36
2	100	240	−140	−70	0.826	−57.85
3	100	144	−44	−22	0.751	−16.53
4	100	108	−8	−4	0.683	−2.73
5	100	108	−8	−4	0.621	−2.48
						Total −215.95

Tax rate 50%

Discount rate 10%

Depreciation - double declining method

Cost of ownership = $1M - $215K = $785K

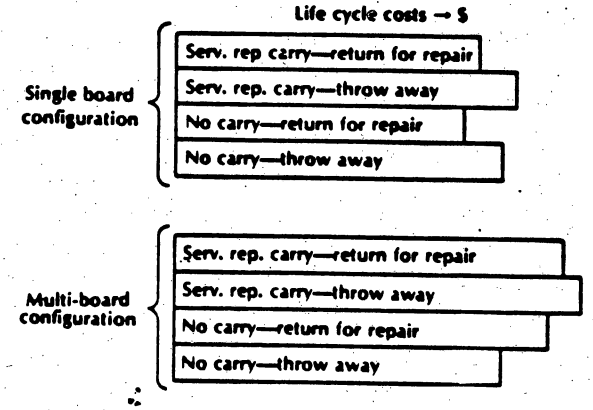


Figure 6–2. Life-cycle costs of alternative configurations.

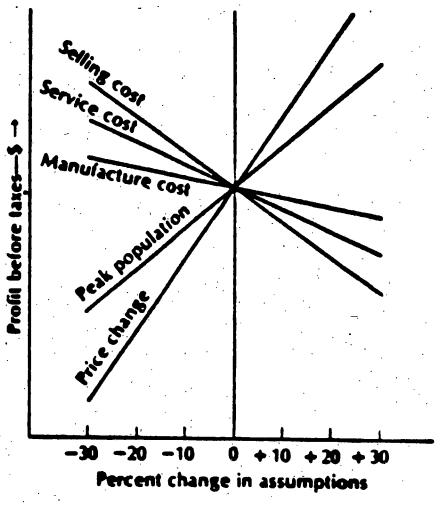


Figure 6–3. Profit sensitivities.

Table 6–5. Cost of manufacturing three systems as a function of time with different MTBFs (base: 6 months, alternative 1: 8.5 months, alternative 2: 10 months) in thousands of dollars.

Quarter	*Shipments*	*Base (SK)*	*Alternative 1 (SK)*	*Alternative 2 (SK)*
1	1	11	11.3	11.8
2	4	44	45.2	47.2
3	73	803	824.9	861.4
4	354	3,894	4,000.2	4,177.2
5	612	6,732	6,915.6	7,221.6
6	820	9,020	9,266.0	9,676.0
7	990	10,890	11,187.0	11,682.0
8	1,000	11,000	11,300.0	11,800.0
9	1,000	11,000	11,300.0	11,800.0
10	1,000	11,000	11,300.0	11,800.0
11	1,000	11,000	11,300.0	11,800.0
12	1,000	11,000	11,300.0	11,800.0
13	1,000	11,000	11,300.0	11,800.0
14	1,000	11,000	11,300.0	11,800.0
15	1,000	11,000	11,300.0	11,800.0
16	1,000	11,000	11,300.0	11,800.0
17	650	7,150	7,345.0	7,670.0
18	447	4,917	5,051.1	5,274.6
19	320	3,520	3,616.0	3,776.0
20	229	2,519	2,587.7	2,702.2

Table 6–6. Cost of maintaining three systems with different MTBFs (base: 6 months, alternative 1: 8.5 months, alternative 2: 10 months) (in thousands of dollars).

Quarter	*Base* (*SK*)	*Alternative 1* (*SK*)	*Alternative 2* (*SK*)
0	17	17	17
1	425	425	425
2	196	177	170
3	554	493	462
4	847	757	709
5	909	818	773
6	1,179	1,045	992
7	1,243	1,075	1,007
8	1,576	1,309	1,239
9	1,777	1,481	1,361
10	1,852	1,527	1,401
11	2,042	1,673	1,521
12	2,194	1,776	1,609
13	2,336	1,876	1,693
14	2,484	1,981	1,782
15	2,626	2,081	1,866
16	2,609	2,044	1,831
17	2,448	1,866	1,651
18	2,345	1,767	1,542
19	2,265	1,677	1,464
20	2,224	1,633	1,419

Table 6–7. Discounted cost of manufacturing and service for the base system (in thousands of dollars).

Quarter	*Manufacturing* ($K)	*Manufacturing'* ($K)	*Service* ($K)	*Service'* ($K)
0			17	17.0
1	11	10.5	425	406.1
2	44	40.2	196	178.9
3	803	700.4	554	483.2
4	3,894	3,245.4	847	705.9
5	6,732	5,360.9	909	723.9
6	9,020	6,863.1	1,179	897.1
7	10,890	7,917.0	1,243	903.7
8	11,000	7,640.9	1,576	1,094.7
9	11,000	7,300.7	1,777	1,179.4
10	11,000	6,975.7	1,852	1,174.4
11	11,000	6,665.1	2,042	1,237.3
12	11,000	6,368.3	2,194	1,270.2
13	11,000	6,084.8	2,336	1,292.2
14	11,000	5,813.8	2,484	1,312.9
15	11,000	5,555.0	2,626	1,326.1
16	11,000	5,307.6	2,609	1,258.9
17	7,150	3,296.4	2,448	1,128.6
18	4,917	2,165.9	2,345	1,033.0
19	3,520	1,481.5	2,265	953.3
20	2,519	1,013.0	2,224	894.4
Present values		89,806.4		19,471.2

Total present value = $109,277.6

Table 6–8. Discounted cost of manufacturing and service for the first alternative with a 42 percent improvement in MTBF (in thousands of dollars).

Quarter	Manufacturing Cost ($K)	Manufacturing Cost' ($K)	Service ($K)	Service' ($K)
0			17	17.0
1	11.3	10.8	425	406.1
2	45.2	41.3	177	161.6
3	824.9	719.5	493	430.0
4	4,000.2	3,333.9	757	630.9
5	6,915.6	5,507.1	818	651.4
6	9,266.0	7,050.3	1,045	795.1
7	11,187.0	8,133.0	1,075	781.5
8	11,300.0	7,849.3	1,309	909.3
9	11,300.0	7,499.8	1,481	982.9
10	11,300.0	7,165.9	1,527	968.3

Table 6–8 –*Continued*

Quarter	Manufacturing Cost ($K)	Manufacturing Cost' ($K)	Service ($K)	Service' ($K)
11	11,300.0	6,846.8	1,673	1,013.7
12	11,300.0	6,542.0	1,776	1,028.2
13	11,300.0	6,250.7	1,876	1,037.7
14	11,300.0	5,972.4	1,981	1,047.0
15	11,300.0	5,706.5	2,081	1,050.9
16	11,300.0	5,452.4	2,044	986.3
17	7,345.0	3,386.3	1,866	860.3
18	5,051.1	2,225.0	1,767	775.7
19	3,616.0	1,521.9	1,677	705.8
20	2,587.7	1,040.6	1,633	656.7
Present values		92,255.7		15,896.6
Total present value = $108,152.3				

Table 6-9. Discounted cost of manufacturing and service for the second alternative with a 67 percent improvement in MTBF (in thousands of dollars).

Quarter	Manufacturing Cost ($K)	Manufacturing Cost' ($K)	Service ($K)	Service' ($K)
0			17	17.0
1	11.8	11.3	425	406.1
2	47.2	43.1	170	155.2
3	861.4	751.4	462	403.0
4	4,177.2	3,481.5	709	590.9
5	7,221.6	5,750.8	773	615.6
6	9,676.0	7,362.3	992	754.8
7	11,682.0	8,492.8	1,007	732.1
8	11,800.0	8,196.6	1,239	860.6
9	11,800.0	7,831.7	1,361	903.3
10	11,800.0	7,483.0	1,401	895.4
11	11,800.0	7,149.8	1,521	921.6
12	11,800.0	6,831.5	1,609	931.5
13	11,800.0	6,527.3	1,693	936.5
14	11,800.0	6,236.7	1,782	941.8
15	11,800.0	5,958.9	1,866	942.3
16	11,800.0	5,693.6	1,831	883.5
17	7,670.0	3,536.1	1,651	761.2
18	5,274.6	2,323.5	1,542	679.3
19	3,776.0	1,589.3	1,464	616.2
20	2,702.2	1,086.7	1,419	570.6
Present values		96,337.8		14,511.5

Total present value = $110,849.3

Table 8-31. Comparison of VAX-11/750 and VAX-11/780 features.

Feature	*VAX-11/750*	*VAX-11/780*	*Benefit*	*Aid MTTF*	*Aid MTTR*
Fault Intolerance					
Air flow	Blowers	Blowers	Lowers chip junction temperature	Yes	No
LSI	Memory chips, 90% of CPU Logic functions implemented as custom gate array	Memory Chips	Fewer chips result in fewer boards, more reliable per function over SSI/MSI, lower power consumption, hence, cooler junction temperatures	Yes	Yes
Cabling	Card cage fixed mounted and not on slides. No internal cables. Connections through backplane, no cables to cards.		Fewer pluggable connectors to fail	Yes	Yes
Physically bounded system	Yes	No	Limited number of system configurations; CPU and memory in one cabinet results in greater control of environmental factors such as temperature and electromagnetic interference.	Yes	Yes
Sensors and indicators	Power loss, temperature, air flow	Power loss	Protects system from damage resulting from emergency conditions	Yes	Yes
Modular power supply	Yes	Yes	Easy replacement	No	Yes
Fault Tolerance					
Main memory	7-bit ECC per 32-bit word	8-bit ECC per 64-bit word	Tolerates transient and permanent failures. Logging of error information allows quick fault isolation.	Yes	Yes

Control store	2 parity bits: 1 even parity, 1 odd parity, over disjoint subfields of the 78-bit-wide control store	3 parity bits, 1 per 32 bits of control store	Provides tolerance of transient errors as well as partial isolation to the failing chip	No	Yes
	Micro-verify		Control store resident check of data paths, registers, and other portions of the system boot path. Ensures proper boot of system if passed.	No	Yes
Translation buffer	2-way set-associative. 4 parity bits for each set: 1 over 15-bit tag and valid bit; 3 over disjoint subfields of 15-bit page frame number, 4-bit protection, and modify bit	2 way set-associative. 6 parity bits for each set: 3 over 16-bit tag, valid, modify, and 4-bit protection; 3 over 21 bits of page-frame number	Provides faulty chip isolation. Tolerates transients by recalculating TB contents. Tolerates permanent failures by disabling one set.	Yes	Yes
Cache	Direct-mapped cache: 5 parity bits, 1 over 12-bit tag and valid bit; 4 over 32 data bits (byte parity)	2-way set-associative. 7 parity bits per set: 3 over 12-bit tag and valid bit; 4 over 32 data bits (byte parity)	Provides faulty chip isolation. Tolerates transients by refetching cache contents. Tolerates permanent failures by disabling cache (11/750) or one set (11/780).	Yes	Yes
Synchronous Backplane Interconnect (SBI)	None	2 parity bits: 1 over 32-bit data/address field, 1 over 12 bits of control information	Detects errors and isolates to faulty bus port. Transients tolerated by bus level retry.	No	Yes
		Silo captures last 16 bus cycles	Isolates faulty chips	No	Yes
Unibus Adapter	None	Parity on data paths and Unibus Map	Provides faulty chip isolation. Transients tolerated by retry.	No	Yes
Massbus	Data and Control Bus lines parity. Data buffer parity.	Data and Control Bus lines parity	Provides faulty chip isolation. Transients tolerated by retry.	No	Yes
Watchdog timer	None	In LSI-11 console processor	Detects hung machine and allows automatic restart	No	No

(Table continues on next page)

Table 8–31 —*Continued*

Feature	*VAX-11/750*	*VAX-11/780*	*Benefit*	*Aid MTTF*	*Aid MTTR*
Clock margining	None	Change clock speed	Aids isolation of timing problems	No	Yes
Maintenance registers	Machine Check Error Summary	SBI Fault/Status	Aids fault isolation	No	Yes
	Cache Error	SBI Silo Comparator			
	Machine Check Status	SBI Error			
		SBI Timeout Address			
		SBI Maintenance Buffer			
		Translation Parity			
Visibility Bus	Internal signals made available to microdiagnostics	Internal signals made available to the console or microdiagnostics	Aids fault isolation	No	Yes
Chip Sockets	Gate array	None	Allows replacement of individual gate array chips	No	Yes
Remote diagnostic module	Load/examine critical machine registers	Load/examine critical machine registers	Provides remote, expert troubleshooting	No	Yes
		Single-step sequencer			
	Monitor control store output for control store verification	Clock margining			
	Error status registers readable over W-bus	Error status registers readable over ID Bus			
	Monitor Memory Data Register (MDR) to verify memory-CPU transfers and opcode undergoing execution	Access to V-Bus			
	Visibility of cache and translation buffer contents to insure correct functionality	Microdiagnostics loadable into writable control store			
	Access to V-Bus. On-board 64-word writable control store for microdiagnostics.				

Table 6–3. Input and output parameters for a maintenance cost model.

Primary data inputs are listed by category as follows:	*Primary data outputs obtained are listed as follows:*
Part data	
1. Unit cost	1. Increase in number of service personnel by year
2. Repair cost	2. Number of spares replaced per year
3. Repair transportation cost	3. Average cost of a spare item
4. Power-on hours	4. Mean corrective maintenance time
5. Repair turnaround time	5. Number of spares returned from the field per year
6. Repair attrition	6. Number of additional spares needed per year
7. Part population	7. Number of spares shipped to the field per year
8. Erroneous replacements	8. Number of nonrepairable parts
9. Replacement rates (MTBF)	9. Number of parts in field inventory
10. Reliability growth	10. Number of parts returned from the repair facility
11. On-site time to repair	11. Initial cost of parts per year
12. Salvage value	12. Initial parts depreciation/tax recovery per year
	13. Cost of replaced spare parts per year
Business economic factors	14. Tax recovery from replaced parts per year
1. Life-cycle period	15. Cost of service labor per year
2. Corporation-selected depreciation	16. Shipping cost of failed parts per year
3. Corporate tax rate	17. Cost of vendor repair of failed parts per year
4. Service Personnel labor rate	18. Shipping cost of spares per year
5. Part cost improvement	19. Salvage value of nonrepair parts
6. Machine placements	
7. Machine workload per service personnel	
Program option controls	
1. Detailed or summarized output	
2. Supplemental quarterly output	
3. Unit cost vs. reliability indifference routine	
4. Part repair or discard evaluation	
5. Service rep carry part or no-carry evaluation	
6. Computations without present value, depreciation, and tax influences	

Table 6–10. A simple life-cycle cost model and its parameters.

Description of Variables:

NUM	Number of equipment items	Input
HOURS	Total life cycle operating hours per equipment item	Input
LABOR	Average labor rate $/MH	50.00
RPPTIP	Ratio of system purchase price to sum of PRICE(I)	0.40
DEVELS	Cost of development	Input
INSTAS	Cost of initial installation material and equipment, $	Input
MTTF	Mean Time To Failure for the whole system	Input
TQUANT	Total quantity of systems to be made to amortize development costs	Input

All the following are arrays of size 'NUM'

PRICE	Initial price	Input
TFAIL	Failure rate for total quantity of equipment item(I), F/MHR	Input
QUANT	Total quantity of equipment item(I)	Input

These are assumed to be constant:

FIXILQ	Fraction of failures repaired at intermediate level (IL) branch office by quantity	0.20
FIXDLQ	Fraction of failures repaired at depot level (DL) by quantity	0.20, 0.50
AILRS	Average IL repair material cost percent of PRICE	0.05
AILRT	Average IL repair time, mh	5.0
ADLRS	Average DL repair material cost percent of PRICE	0.10
ADLRT	Average DL repair time, mh	5.0
AOLRT	Average organizational level (OL) factory repair time, mh	5.0
AOLRTS	Average OL repair material cost percent of PRICE	0.20
TRIP	Cost to make service trip	100

Equations:

Acquisition = $RPPTIP \times \sum |PRICE(I) \times QUANT(I)|$

Development = $DEVELS/TQUANT$

Initial Installation = $INSTAS$

Initial and replaceable spares = $(0.05 + HOURS \times 6.0E\text{-}6) \times Acquisition$

Organizational level maintenance = $int(HOURS/MTTF) \times (\sum |TFAIL(I) \times MTTF \times 1.0E\text{-}6 \times (1.0 - FIXDLQ - FIXILQ) \times X(I)| + TRIP)$
Where $X(I) = AOLRTS \times PRICE(I) + AOLRT \times LABOR$

Intermediate and depot level maintenance = $int(HOURS/MTTF) \times (\sum |TFAIL(I) \times MTTF \times 1.0E\text{-}6 \times (FIXILQ \times V(I) + FIXDLQ \times W(I)| + TRIP)$
Where $V(I) = AILRS \times PRICE(I) + AILRT \times LABOR$
and $W(I) = ADLRS \times PRICE(I) + ADLRT \times LABOR$

Inventory management = $4.0 \times Initial\ Installation$

Support Equipment = 0

Personnel Training = 0

Management and Technical Data = 0

New Facilities = 0

Source: Note this model is a modified version of one proposed in *Life Cycle Analysis Utilizing Generalized Data Elements* by Susan Eamon and Al Spann of GTE Sylvania Incorporated.

Table 6–4. A feature failure-mode matrix.

	Percent of System Failures	Data Path Loop-Around	Error Simulation	Internal Parity	Speed Check	Skew Check	Power Check	Reel LEDs
Permanent data path	11%	74	24	74				
Intermittent data path	2			168				
Controller	6							
Error logic	6	24	54	24				
Head preamp	10	84		84				
Servos	25				390	210		66
Power supply	5						90	
Minutes saved		18	6	21	97.5	52.5	4.5	16.5

Module: AUP   Page: 1   Comments: 7.541 / 9.955

NAME-ID-FN	Complex.	Type	Qty.	Det. Pr.	F.R.		
VA [illegible] Reid @0.102	-	H175 H175 H105	15F/5 1F/5 1F/5	1.0 0.5 0.5	0.031 0.031 0.017	0.103	0.098
VA Latch VB<31, A:9> (VA Forced)	-	schematic	1	0.124 0.248	0.248	0.248	0.124
TBF	-	schematic	1	3.40 3.45	3.45	3.45	3.401
VA PTY CD	--	H107	2F/3	1.0	0.018	0.012	0.012
SLC PA PTY ERR		H160 H175 H104	2 2F/5 1F/4	1.0 ↓	0.0 1 0.031 0.018	0.059	0.059
VA PTY ERR OUT		H105 H175	1F/4 1F/5	1.0 ↓	0.017 0.031	0.010	0.010
TBL4	-	LF MCA	1	132 282	0.282	0.282	0.121
PA Latch		H175	2F/4	1.0	0.029	0.218	0.218

4.043 / 4.382

Figure 1. Filled-in Worksheet

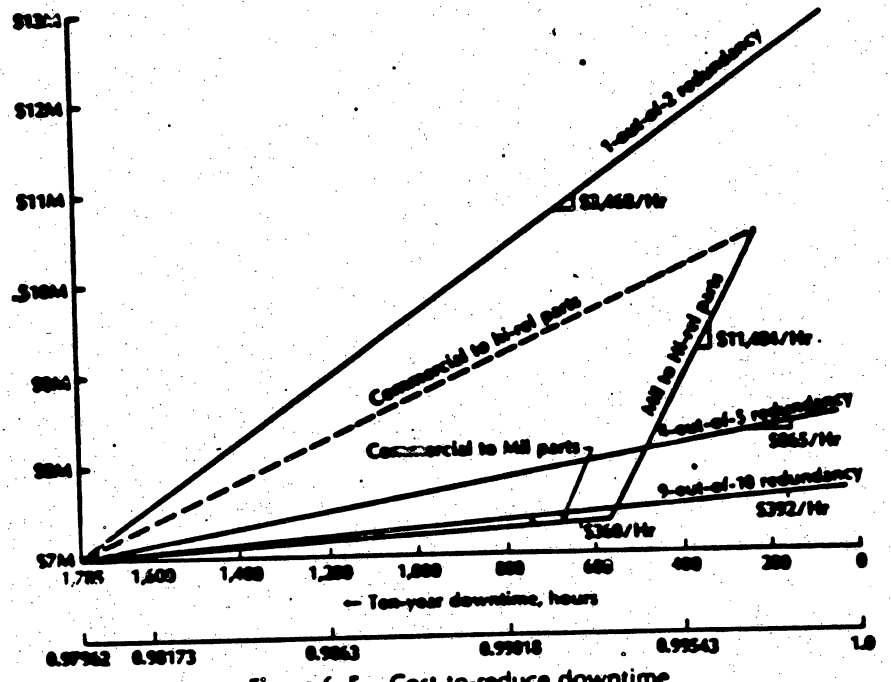


Figure 6–5. Cost-to-reduce downtime.

Table 6–11. Example of effects of component quality levels on LCC.

Parts Quality Grade: *Total LRUs:* *Availability:* *Mean Uptime:*	*Commercial* *1,600* *0.979624* *24 Hrs.*	*MIL* *1,600* *0.993635* *78 Hrs.*	*HI-REL* *1,600* *0.997444* *195 Hrs.*
Acquisition	$ 2,560,000	$ 2,988,244	$ 4,400,516
Development	207,999	207,999	207,999
Initial installation	84,979	84,979	84,979
Initial and replacement spares	2,880,000	3,262,804	4,950,596
Organization level maintenance	125,974	122,017	120,968
Branch and depot level maintenance	430,346	147,256	84,849
Inventory management	309,287	309,287	309,287
Support equipment	185,000	185,000	185,000
Personnel training	0	0	0
Management and technical data	200,804	196,638	195,530
New facilities	0	0	0
Life-cycle cost	$ 6,984,389	$ 7,416,224	$ 10,539,720
		Cost to Avoid Downtime $360/Hr.	Cost to Avoid Downtime $11,484/Hr.

Table 6-12. Example of effects on LCC of reliability improvement via redundancy.

	A	B 9-out-of-10 Redundancy		
Configuration:	Series	11% Redundancy	63% Redundancy	100% Redundancy
Total LRUs:	1600	1617	1706	1777
Availability:	0.979624	0.981619	0.991583	0.999540
Mean uptime:	24 Hrs.	26 Hrs.	58 Hrs.	1.085 Hrs.
Acquisition	$ 2.560.000	$ 2.588.423	$ 2.730.836	$ 2.844.423
Development	207.099	216.412	225.545	235.674
Initial installation	84.979	85.626	85.619	86.046
Initial and replacement spares	2.880.000	2.908.423	3.050.636	3.164.423
Organization level maintenance	125.974	127.370	134.372	139.970
Branch and depot level maintenance	430.346	435.125	459.034	478.163
Inventory management	309.287	309.287	309.287	309.287
Support equipment	185.000	185.000	185.000	185.000
Personnel training	0	0	0	0
Management and technical data	200.804	202.291	209.710	215.649
New facilities	0	0	0	0
Life-cycle cost	$ 6.984.389	$ 7.057.421	$ 7.389.939	$ 7.658.635
		← Cost to avoid downtime $392/Hr. →		

C *4-out-of-5 Redundancy*			D *1-out-of-2 Redundancy*		
12% Redundancy	69% Redundancy	100% Redundancy	18% Redundancy	79% Redundancy	100% Redundancy
1640	1839	2000	1760	2560	3200
0.981619	0.991562	0.999484	0.981616	0.991453	0.999186
26 Hrs.	58 Hrs.	967 Hrs.	26 Hrs.	58 Hrs.	613 Hrs.
$ 2,624,000	$ 2,943,906	$ 3,199,903	$ 2,816,000	$ 4,095,896	$ 5,119,392
216,527	226,437	237,812	217,151	231,849	249,533
85,219	86,419	87,379	85,939	90,739	94,579
2,944,000	3,263,906	3,519,903	3,136,000	4,415,898	5,439,892
129,122	144,869	157,466	138,571	201,557	251,946
441,110	494,985	537,931	473,384	688,559	860,683
309,287	309,287	309,287	309,287	309,287	309,287
185,000	185,000	185,000	185,000	185,000	185,000
0	0	0	0	0	0
204,143	228,845	234,289	214,167	289,971	334,412
0	0	0	0	0	0
$ 7,138,498	$ 7,875,624	$ 8,468,890	$ 7,575,499	$10,498,950	$12,845,220
← Cost to avoid downtime $865/Hr. →			← Cost to avoid downtime $3,468/Hr. →		

USING FIELD SERVICE LCBR

MAKING BUSINESS TRADEOFFS FOR RELIABILITY

MTBPR : MEAN-TIME-BETWEEN-PARTS-REPLACEMENT

F.S. BUSINESS PLAN GOAL = 2000 HRS

CPU LOGIC MODULES

USING MIL217D - LAMBDA

PREDICTING THE IN-FIELD RELIABILITY

ASSUMES MATURE ENGINEERING/MANUFACTURING

REVIEWING RELIABILITY "LEVERS"

CALCULATING RELIABILITY IMPACT OF DESIGN CHANGES

DOES NOT DIRECTLY PREDICT INTERMITTENT/TRANSIENT RATES

0 TECHNOLOGY SELECTION

LSI VERSUS MSI

LSI: 3 - 10 TIMES AS RELIABLE AS MSI

VENDOR RELIABILITY ESTIMATES (SPECS)

VENDORS SHOULD BEAT MIL217D VALUES

	MIL	VEND
ECL LSI	.53	.36
ECL MSI	.022	.012
ECL RAM	.25	.06

AVERAGE FAILURE RATE PER MILLION HOURS

- VENDOR SCREENING SELECTION

  "STRESS SCREENING"

  "PRECONDITIONING"

  "BURNED-IN"

  EARLY DETECTION OF PATENT/LATENT DEFECTS

  DETECTION OF "MAVERICK" LOTS

  COMMERCIAL "HI-REL" PARTS

  10¢/part factor of 2-4 better

  100% THERMAL CYCLE

  100% BURN-IN

  100% HI/LOW TEMP TESTING

- TECHNOLOGY QUALIFICATION
  - DEFINED IN PURCHASE SPECIFICATION
    - USUALLY REFERENCING MIL883 METHODOLOGY
  - DETAILED FAILURE DATA ON CHOSEN TECHNOLOGY(S)
  - INITIAL VERIFICATION OF COMPONENT RELIABILITY
  - POSSIBILITY FOR PERIODIC REQUA_'S

0 JUNCTION TEMPERATURE REDUCTION

$T_J = T_{AMB} + (P_{WATTS} \times R_{DA})$

FOR A 10 DEGREE INCREASE IN MCA JUNCTION TEMP:

44% INCREASE IN FAILURE RATE - VENDOR

32% INCREASE IN FAILURE RATE - MIL217D

THERMAL RESISTANCE (R_{DA}) IMPROVEMENTS

CERAMIC PARTS

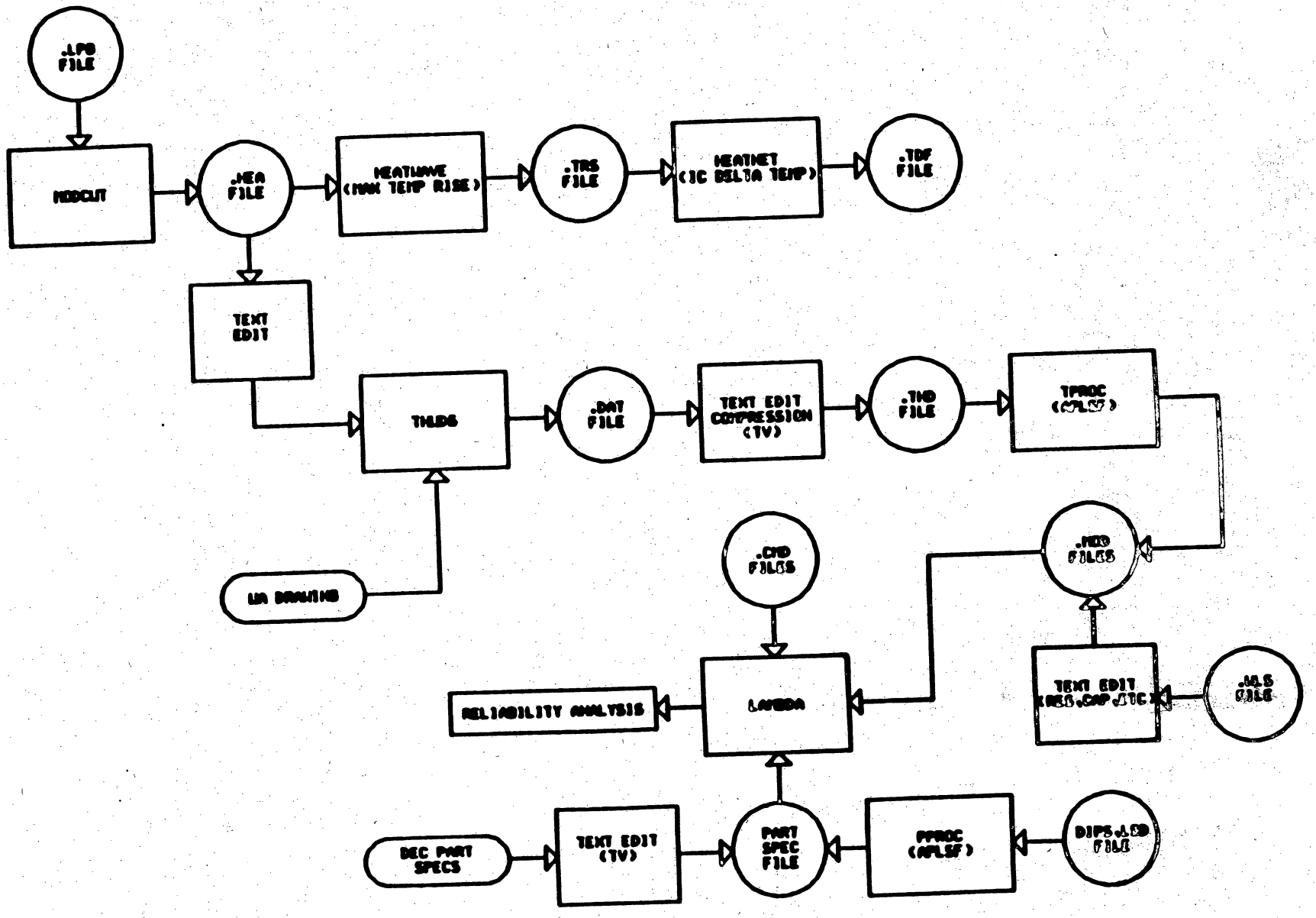
HEAT SINKS

AIR FLOW

"SYSTEM" LEVEL THERMAL ANALYSIS - THUDS

R-

VENUS THERMAL / RELIABILITY DESIGN PROCESS



DRN.	DATE	ENG.	DATE	TITLE: VENUS THERMAL RELIABILITY DESIGN	
CHK'D.	DATE			SIZE C	CODE CS	REV. A

CONCLUSION

CPU LOGIC CONFIGURATION

F.S. MINIMUM ACCEPTABLE GOAL	=	2000 HRS.
SPECIFICATION/PREDICTION	=	2726 HRS.
FINAL RQT RESULTS	=	3500 HRS. @90%

4.2) Fault Handling Techniques and Their Evaluation

Reliability bottleneck: (present in all schemes using redundancy)

Classic dilemma: who shall guard the guardians?

Single points of failure

in duplication, the comparison elements

can use redundancy to further reduce single points of failure; but there is a decreasing return for incremental changes

- avoid overconcern about single points of failure: too much redundancy can be less reliable.

Another problem

Synchronization

→ cause of failure

→ cause of perf. degr.

→ solution often taken (common clock) creates single point of failure)

Common to most replication techniques

Detection using redundancy
to detect failure occurrence,
preferably before error propagates

Fail-Safe if failure occurs,
output is in "safe" position
e.g. bomb fuze
traffic light

Fault-Secure output can be wrong,
but it must be accompanied by
error signal if it is.

Table 3–14. Critical and subcritical input faults for some common logic gates.

Gate Type	*Critical Faults*	*Subcritical Faults*
AND	$1 \rightarrow 0$	$0 \rightarrow 1$
OR	$0 \rightarrow 1$	$1 \rightarrow 0$
NOT	$0 \rightarrow 1$, $1 \rightarrow 0$	None
NAND	$1 \rightarrow 0$	$0 \rightarrow 1$
Majority	None	$0 \rightarrow 1$, $1 \rightarrow 0$

Fail Safe & Fault Secure Logic

Definitions:

Self-checking circuit is one whose outputs are coded in an error code.

Self-testing circuit is one for which, for every fault f from a specific set of faults F, the circuit produces a non code space output for at least one code space input.

Fault-secure circuit is ckt which for every fault $f \in F$ produces either a non code space output or the correct code space output for all code space inputs.

also Totally & Partially self-checking ckts

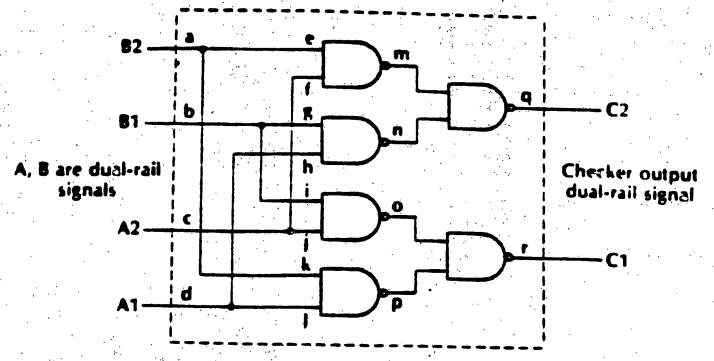


Figure 3–36. Logic circuit of basic TSC comparison element.

Table 3–12. TSC dual-rail comparator responses to stuck-at-faults.

Inputs		*Normal*	*Outputs C2C1 Resulting from Single Stuck-at-1 Faults*																	
B2B1	*A2A1*	*Output*	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*	*k*	*l*	*m*	*n*	*o*	*p*	*q*	*r*
01	01	10	11	10	11	10	10	10	10	10	10	11	11	10	10	00	10	10	10	11
01	10	01	11	01	01	11	11	01	01	11	01	01	01	01	01	01	00	01	11	01
10	01	01	01	11	11	01	01	11	11	01	01	01	01	01	01	01	01	00	11	01
10	10	10	10	11	10	11	10	10	10	10	11	10	10	11	00	10	10	10	10	11
Inputs		*Normal*	*Outputs C2C1 Resulting from Single Stuck-at-0 Faults*																	
B2B1	*A2A1*	*Output*	*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*	*k*	*l*	*m*	*n*	*o*	*p*	*q*	*r*
01	01	10	10	00	10	00	10	10	00	00	10	10	10	10	10	10	11	11	00	10
01	10	01	01	00	00	01	01	01	01	01	00	00	01	01	11	11	01	01	01	00
10	01	01	00	01	01	00	01	01	01	01	01	01	00	00	11	11	01	01	01	00
10	10	10	00	10	00	10	00	00	10	10	10	10	10	10	10	10	11	11	00	10

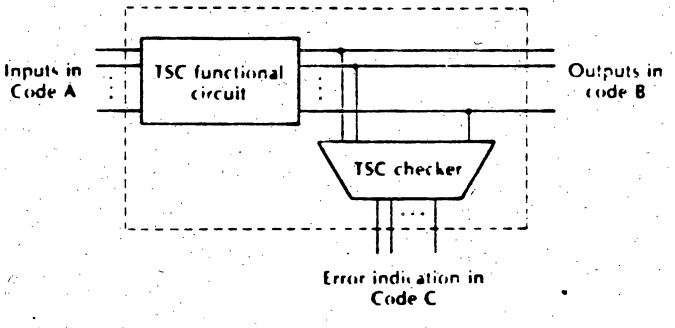


Figure 3–37. A TSC network made from TSC elements

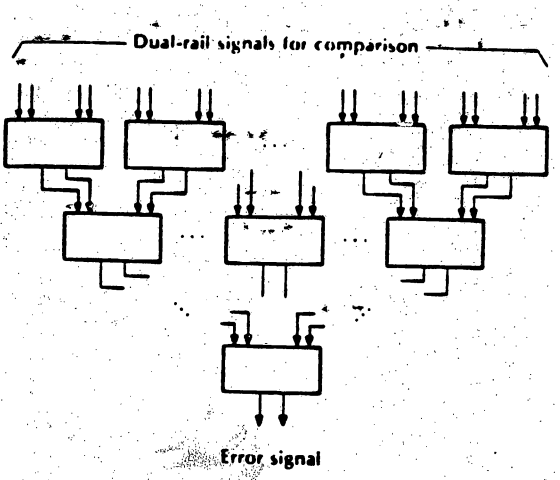


Figure 3–38. Assembly of *n*-input dual-rail signal comparison checker from basic two-input elements.

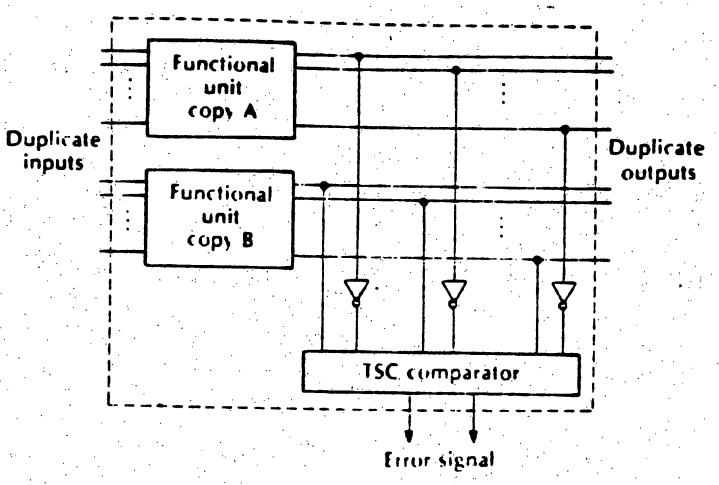


Figure 3–39. TSC network based on duplication as a code.

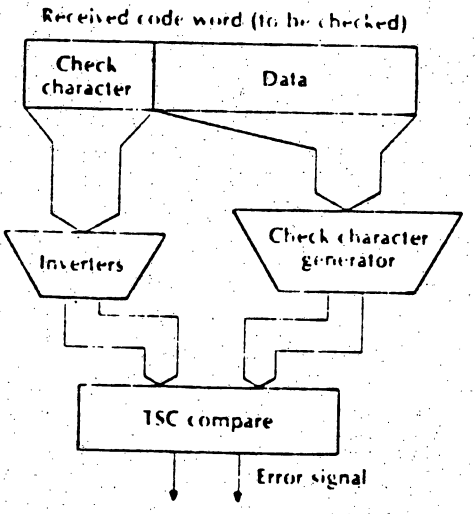


Figure 3–40. TSC detector for separable code based on a TSC comparator.

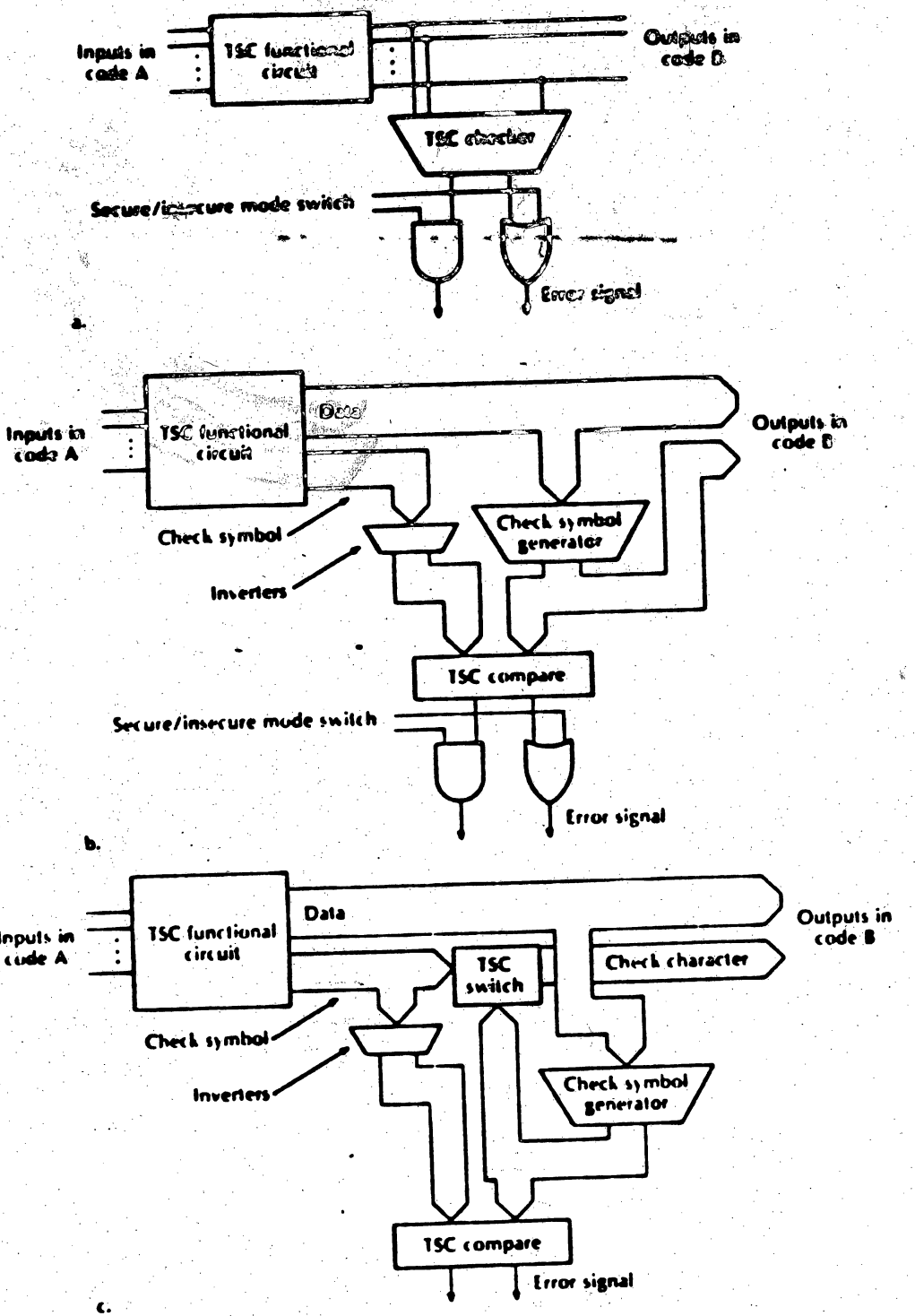


Figure 3–41. Types of PSC networks. a.) Type 1. b.) Type 2 c.) Type 3.

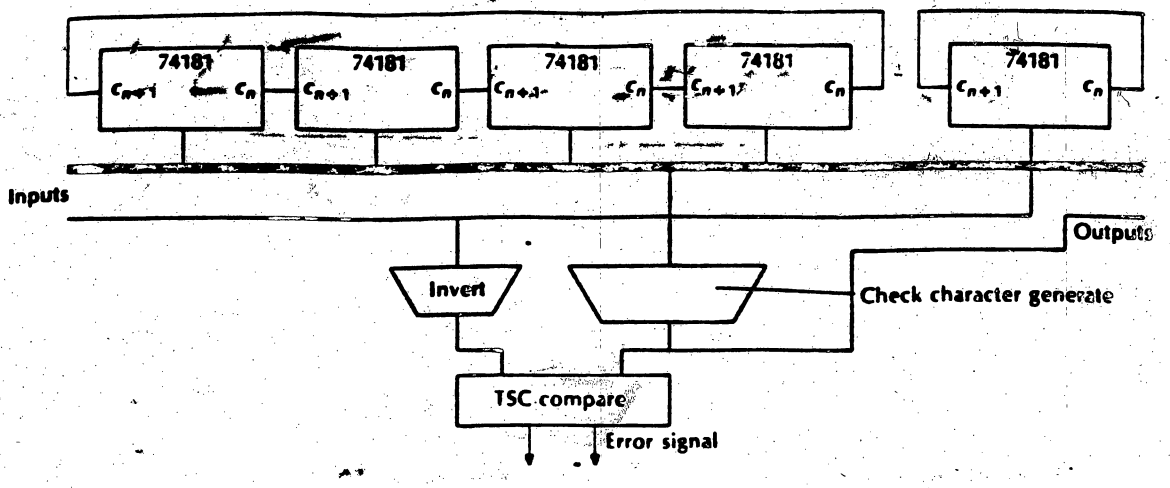


Figure 3–42. Partially self-checking ALU for residue-15 coded operands (Type 2 PSC network).

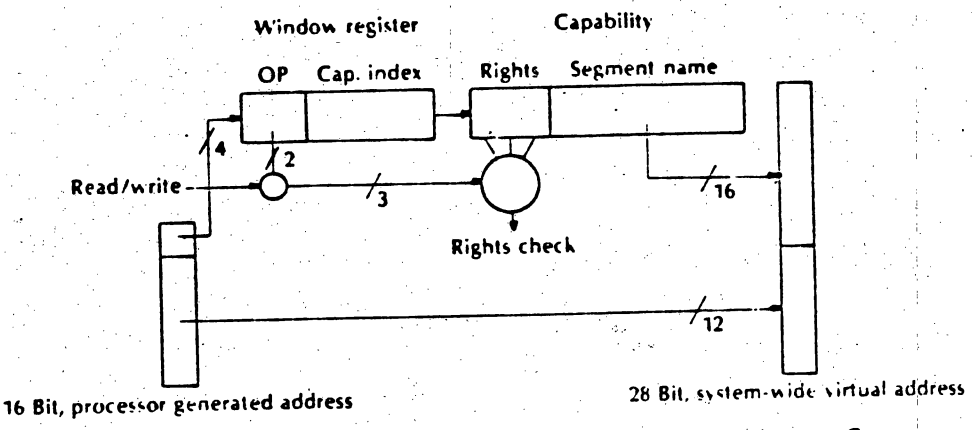


Figure 3–43. Virtual address calculation with capability checking in Cm*.

FAULT TOLERANCE

Goal:

To actually withstand (tolerate) faults, and continue operation.

Static
Dynamic
} Redundancy

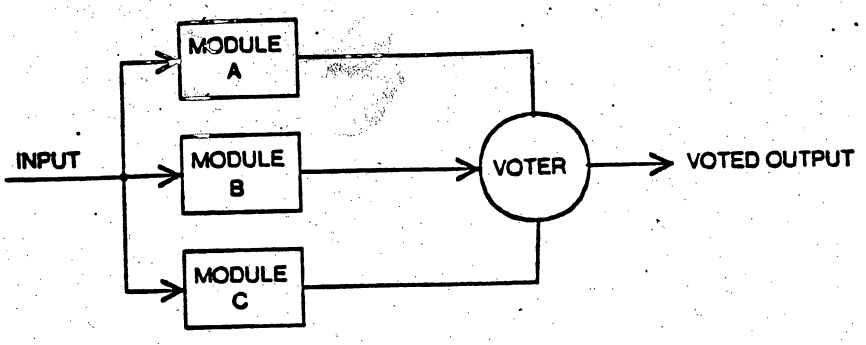


Figure <tmr basic> Basic Triple Moodular Redundancy. Modules A, B, and C are functionally identical.

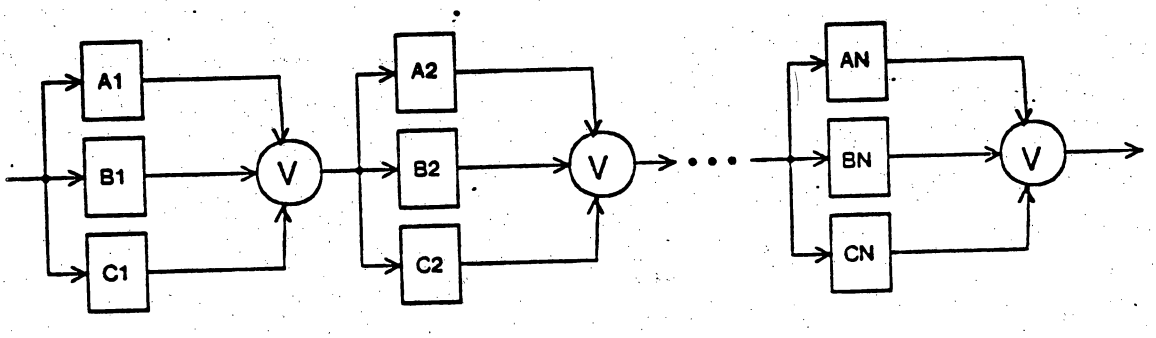


Figure <TMR cascade> Cascading of TMR modules.

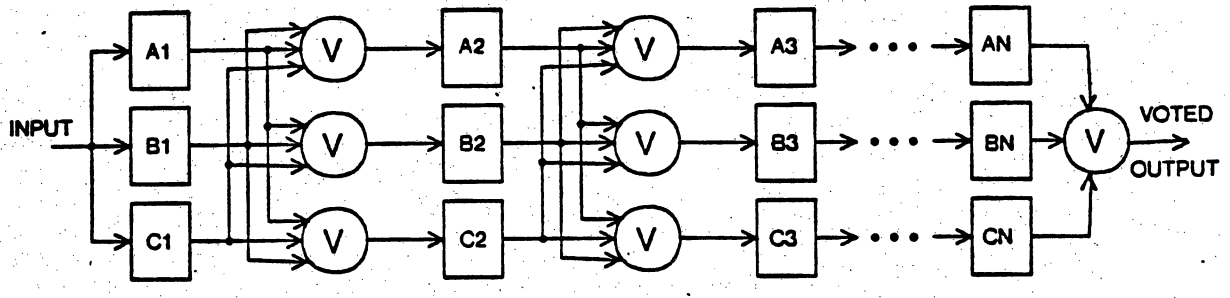


Figure <TMR voters> TMR voters remove almost all single points of failure (except the final output voter).

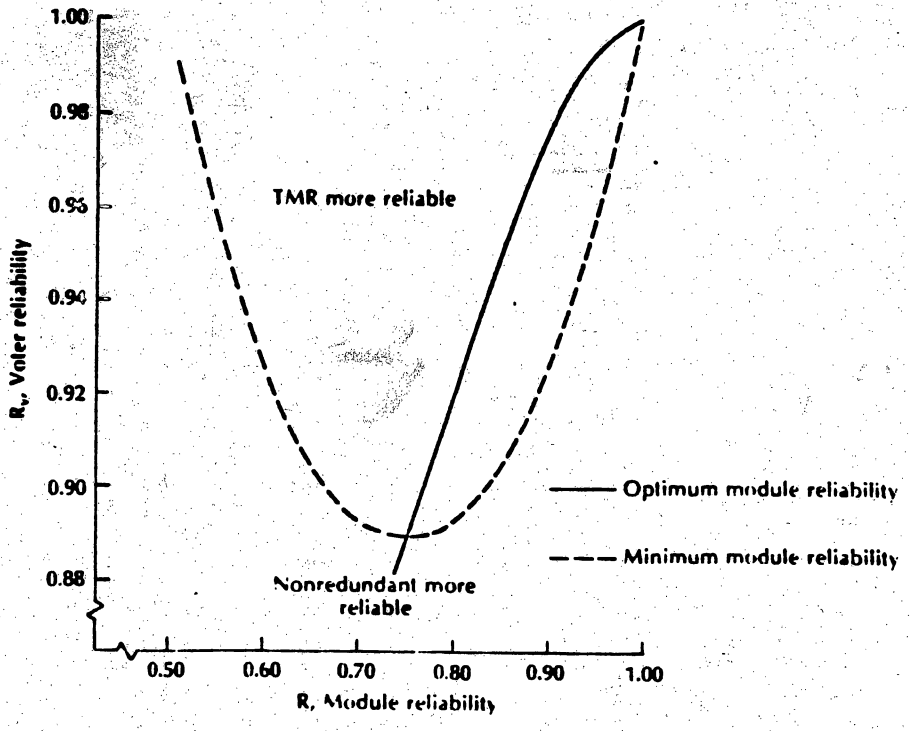


Figure 3–47. Decision regions for single voter TMR. (© 1965 IEEE.)

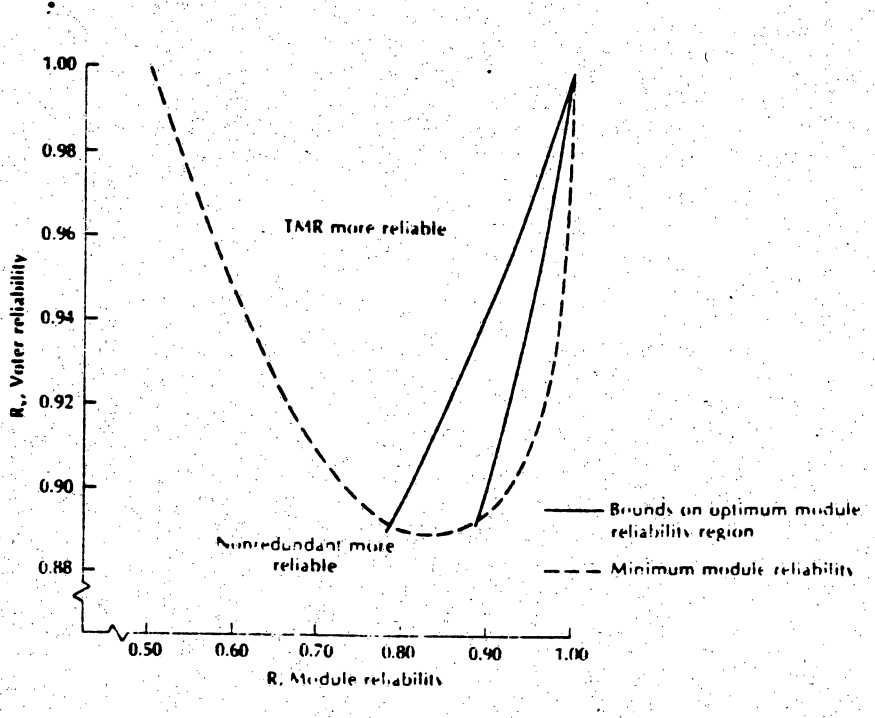
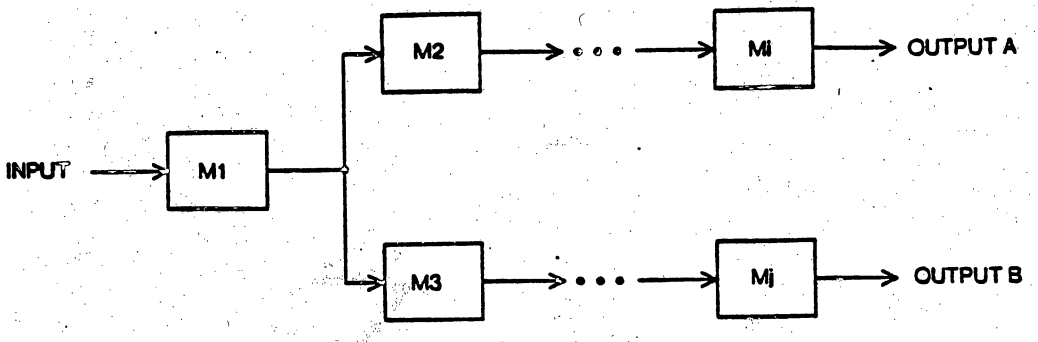


Figure 3–48. Decision regions for triplicated voter TMR. (© 1965 IEEE.)



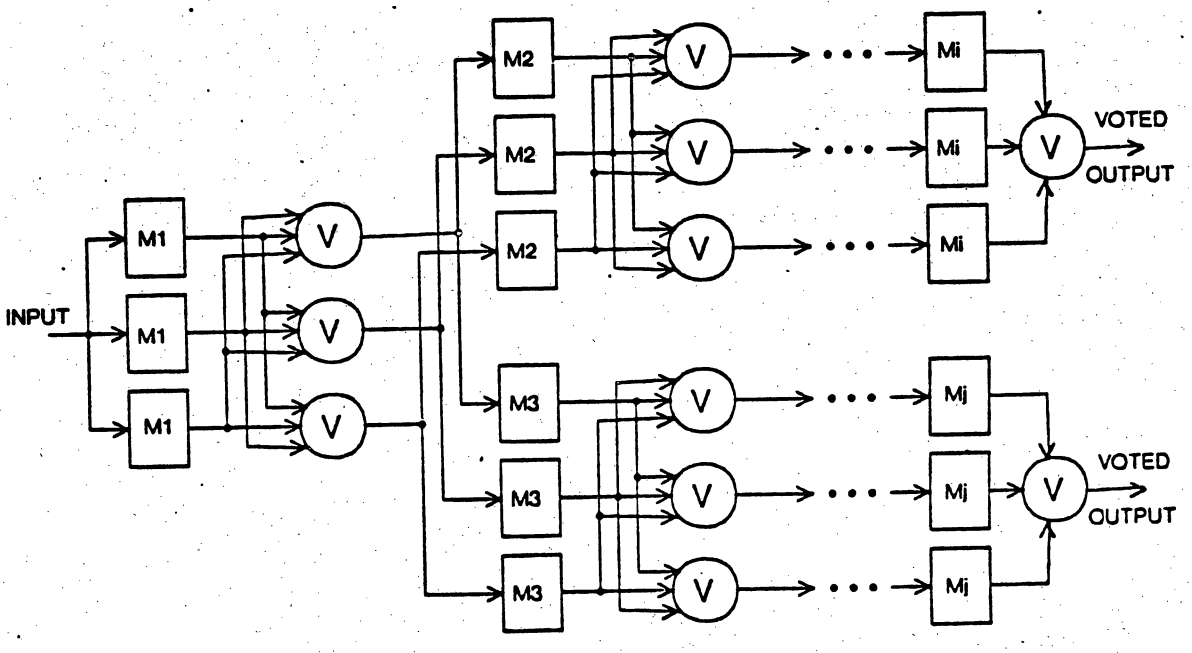


Figure <complex tmr b> TMR can be applied to more complex networks.

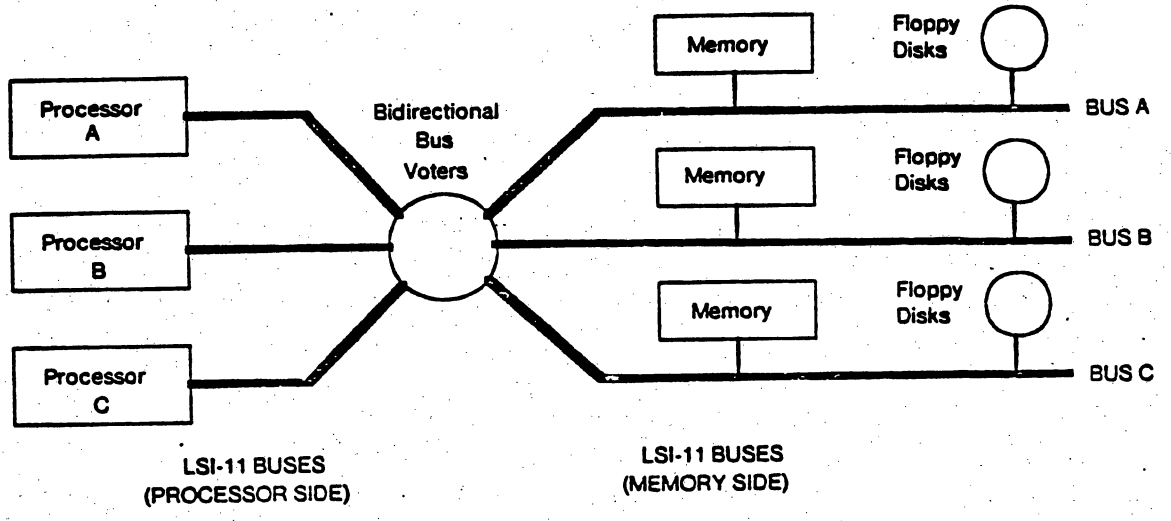


Figure <c.vmp> Basic structure of C.vmp

Combinatorial Model

For a system with N identical modules, K of which are necessary for operation

$$R_{sys}(t) = \sum_{i=0}^{N-K} \binom{N}{i} \left(R_m(t)\right)^{N-i} \left[1 - R_m(t)\right]^i$$

EXAMPLE: TMR system

3 modules, need 2

$$R_{TMR} = \sum_{i=0}^{1} \binom{3}{i} R_m^{3-i} (1-R_m)^i$$

$$= \binom{3}{0} R_m^3 + \binom{3}{1} R_m^3 (1-R_m)$$

$$= R_m^3 + 3R_m^2 (1-R_m)$$

$$= R_m^3 + 3R_m^2 - 3R_m^3$$

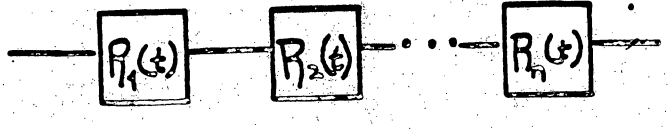
$$R_{TMR} = 3R_m^2 - 2R_m^3$$

which is a very well known formula

$$R_{TMR} = 3e^{-2\lambda t} - 2e^{-3\lambda t}$$

$$\neq K_2 e^{-K_1 t}$$

SERIES/PARALLEL REDUNDANCY



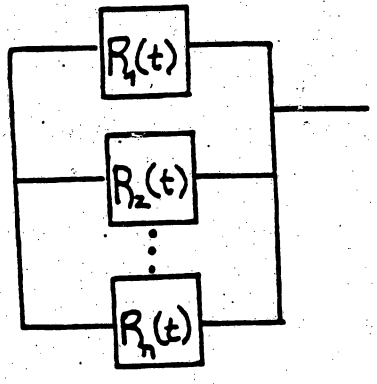
series connection

series reliability unreliability

$$R_{series}(t) = \prod_{i=1}^{n} R_i(t)$$

where $Q_i = 1 - R_i$

$$Q_{series}(t) = 1 - \prod_{i=1}^{n} (1 - Q_i(t))$$



parallel connection

parallel reliability unreliability

$$R_{parallel}(t) = 1 - \prod_{i=1}^{n} (1 - R_i(t))$$

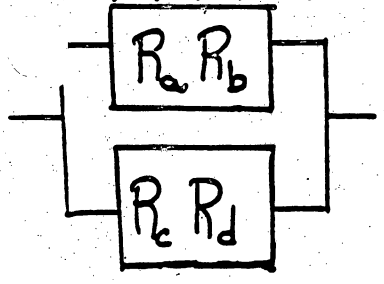
$$Q_{parallel}(t) = \prod_{i=1}^{n} Q_i(t)$$

SERIES/PARALLEL EXAMPLE

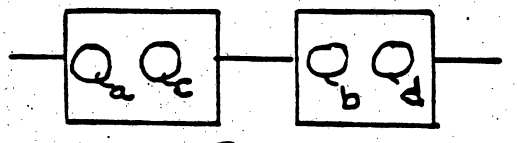
working path includes at least 1 good unit

shorts tolerated
a,b,c,d,ac,ad,bc,bd
opens tolerated
a,b,c,d,ab,cd

shorts tolerated
a,b,c,d,ac,bd
opens tolerated
a,b,c,d,ab,ad,cb,cd



R_{short}



R_{open}

$$R_{short} = 1 - (1-R_aR_b)(1-R_cR_d)$$
$$= R_aR_b + R_cR_d - R_aR_bR_cR_d$$

$$R_{open} = (1 - (1-R_a)(1-R_c))(1 - (1-R_b)(1-R_d))$$
$$= R_aR_b + R_bR_c + R_cR_d + R_aR_d$$
$$+ R_aR_bR_cR_d - R_aR_bR_c$$
$$- R_aR_bR_d - R_aR_cR_d - R_bR_cR_d$$

SERIES/PARALLEL EXAMPLE (cont.)

simplification #1

$$R_a = R_c = R_p,\ R_b = R_d = R_m$$

$$R_{short} = 2R_pR_m \cdot (R_pR_m)^2$$

$$R_{open} = (R_p + R_m)^2 + (R_pR_m)^2 \cdot 2R_pR_m(R_p + R_m)$$

simplification #2

$$R_a = R_b = R_c = R_d = R$$

$$R_{short} = 2R^2 \cdot R^4$$

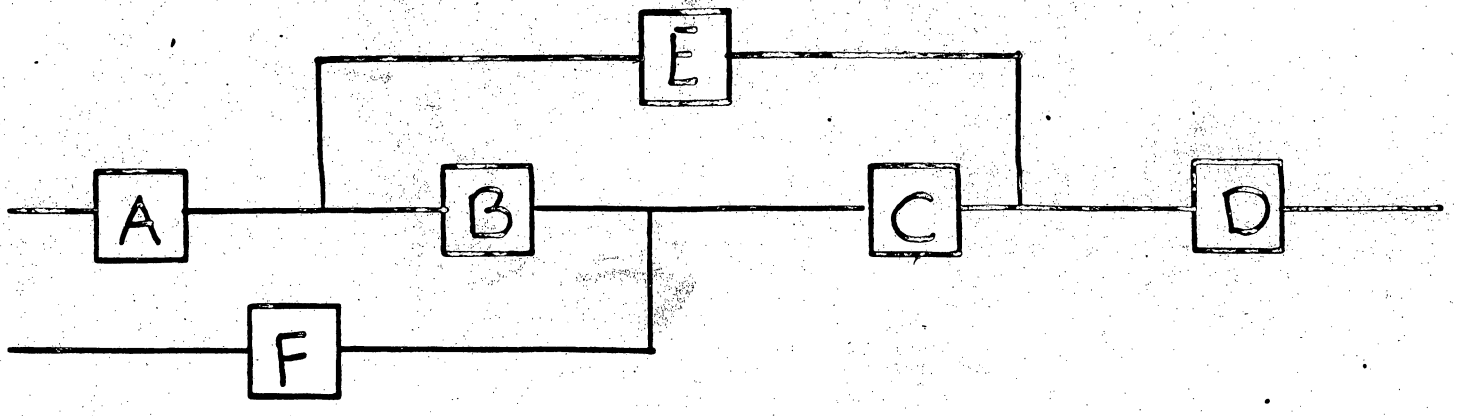
$$R_{open} = 4R^2 \cdot 4R^3 + R^4$$

NOTE:

$$R_{open} > R_{short}$$

for $0 < t < \infty$

REDUCING NONSERIES/NONPARALLEL (1)



working paths: ABCD, AED, FCD

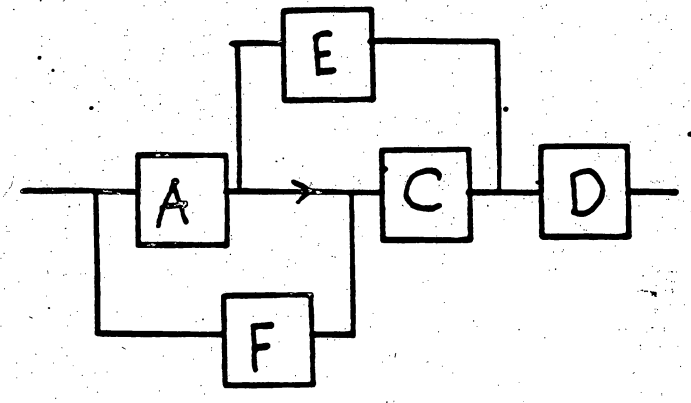
REDUCING NONSERIES/NONPARALLEL (2)

An exact expression for R_{sys} can be derived by expanding around a single module m,

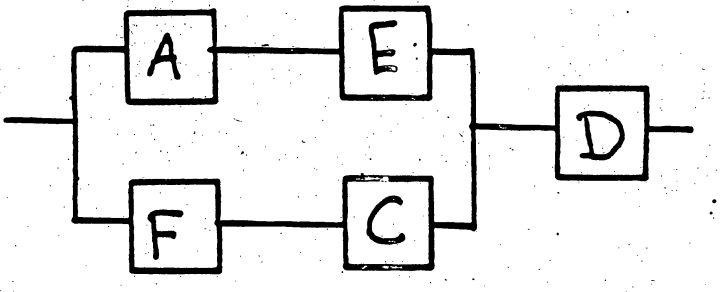
$$R_{sys} = R_m \times \Pr[\text{system works} \mid m \text{ works}] + (1-R_m) \times \Pr[\text{system works} \mid m \text{ fails}] \quad (2)$$

expand example around B

B works

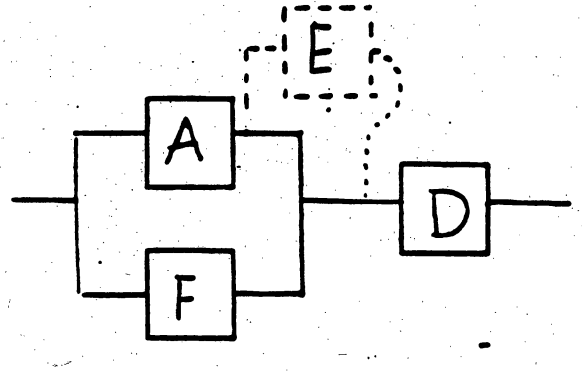


B fails



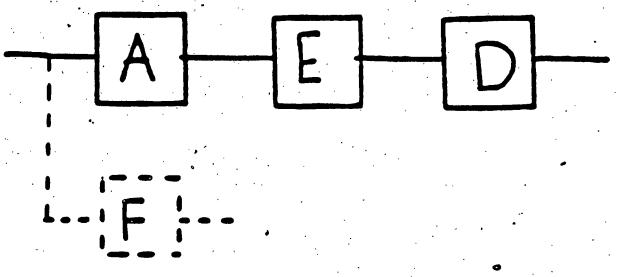
this needs to be expanded again (around C)

B works, C works



ignore E - "shorted out"

B works, C fails



ignore F - leads to "open"

REDUCING NONSERIES/NONPARALLEL (2 cont.)

$$R_{sys} = R_B \times [R_C \times [1-(1-R_A)(1-R_F)]R_D +$$

$$(1-R_C) \times R_A R_E R_D] +$$

$$(1-R_B) \times [1-(1-R_A R_E)(1-R_C R_F)]R_D$$

$$R_{sys} = R_A R_B R_C R_D R_E R_F + R_A R_B R_C R_D + R_A R_D R_E + R_C R_D R_F - R_A R_B R_C R_D R_E -$$

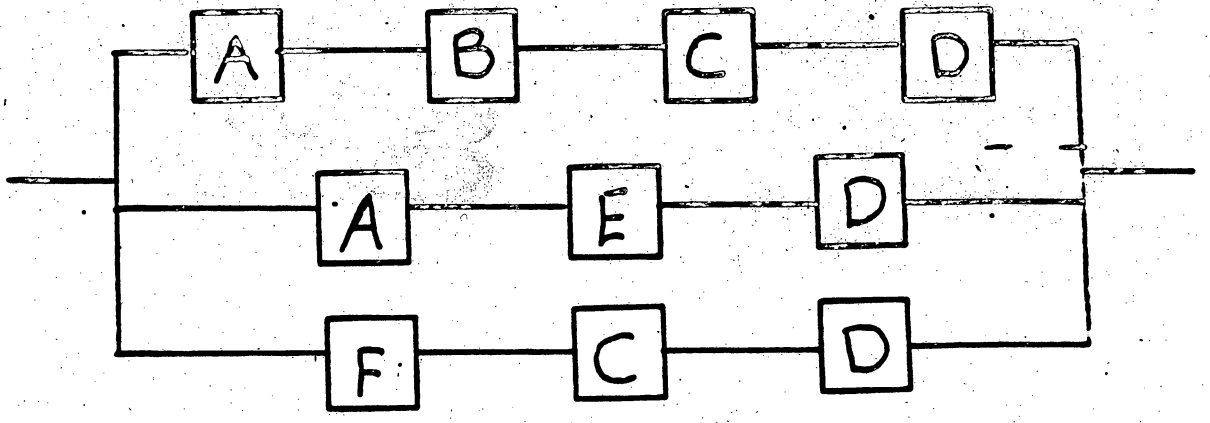
$$R_A R_B R_C R_D R_F - R_A R_C R_D R_E R_F$$

if $R_A = R_B = R_C = R_D = R_E = R_F = R$,

$$R_{sys} = R^6 - 3R^5 + R^4 + 2R^3$$

REDUCING NONSERIES/NONPARALLEL (3)

Reliability Block Diagram



(3) $R_{sys} \approx 1-(1-R_AR_BR_CR_D)(1-R_AR_DR_E)(1-R_CR_DR_E)$

(3.1) $= R_AR_DR_E + R_CR_DR_F + R_AR_BR_CR_D +$

$R_A^2R_BR_C^2R_D^3R_ER_F - R_AR_CR_D^2R_ER_F -$

$R_AR_BR_C^2R_D^2R_F - R_A^2R_BR_CR_D^2R_E$

if $R_A = R_B = R_C = R_D = R_E = R_F = R$,

$R_{sys} \approx 2R^3 + R^4 + R^{10} - R^6 - 2R^7$

this is a ~~lower~~ upper bound for R_{sys}

REDUCING NONSERIES/NONPARALLEL (3 cont.)

the preceding method can be modified to yield the exact result for R_{sys}: in equation 3.1, replace all occurrences of R_m^i with R_m $(i > 0)$

$$(3.1') \quad R_{sys} = R_A R_D R_E + R_C R_D R_F + R_A R_B R_C R_D +$$

$$R_A R_B R_C R_D R_E R_F - R_A R_B R_C R_D R_E -$$

$$R_A R_B R_C R_D R_F - R_A R_C R_D R_E R_F$$

if $R_A = R_B = R_C = R_D = R_E = R_F = R$,

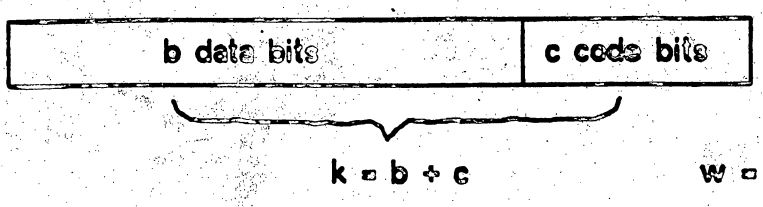
$$R_{sys} = 2R^3 + R^4 - 3R^5 + R^6$$

which is the same result as obtained by the method of expansion

TWO ALTERNATIVES FOR ECC

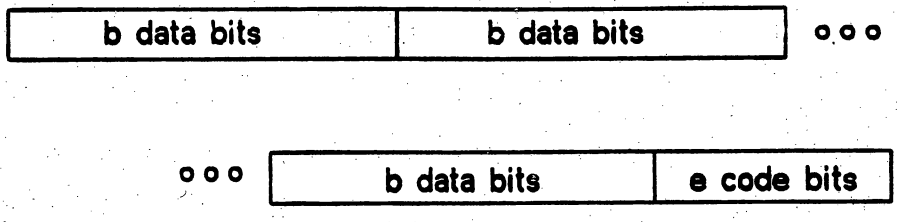
1. Hamming code

a)



SINGLE LOGICAL WORD PER PHYSICAL WORD

b)



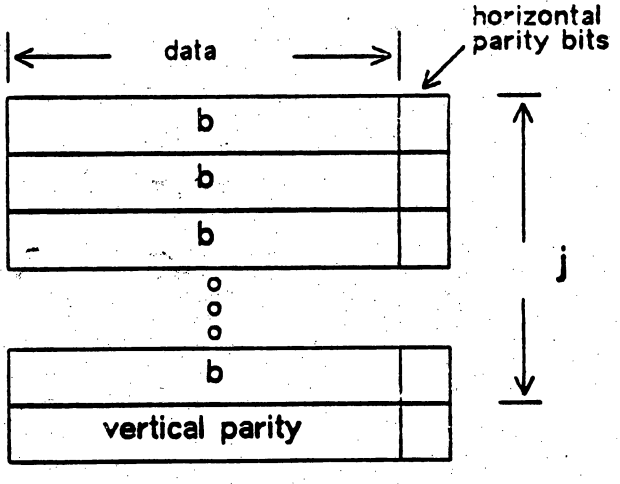
j LOGICAL WORDS PER PHYSICAL WORD

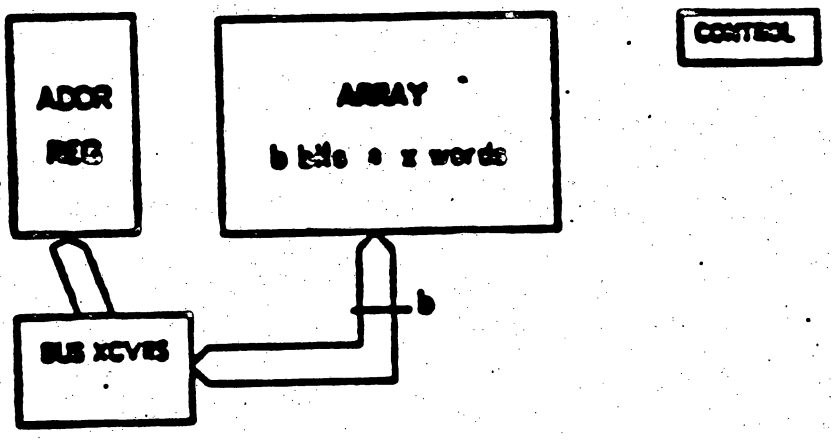
- $e < j * c$
- $k = j*b + c$
- $w = x/j$

2. Block code

$$k = (j + 1) *(b + 1)$$

$$w = x/j$$

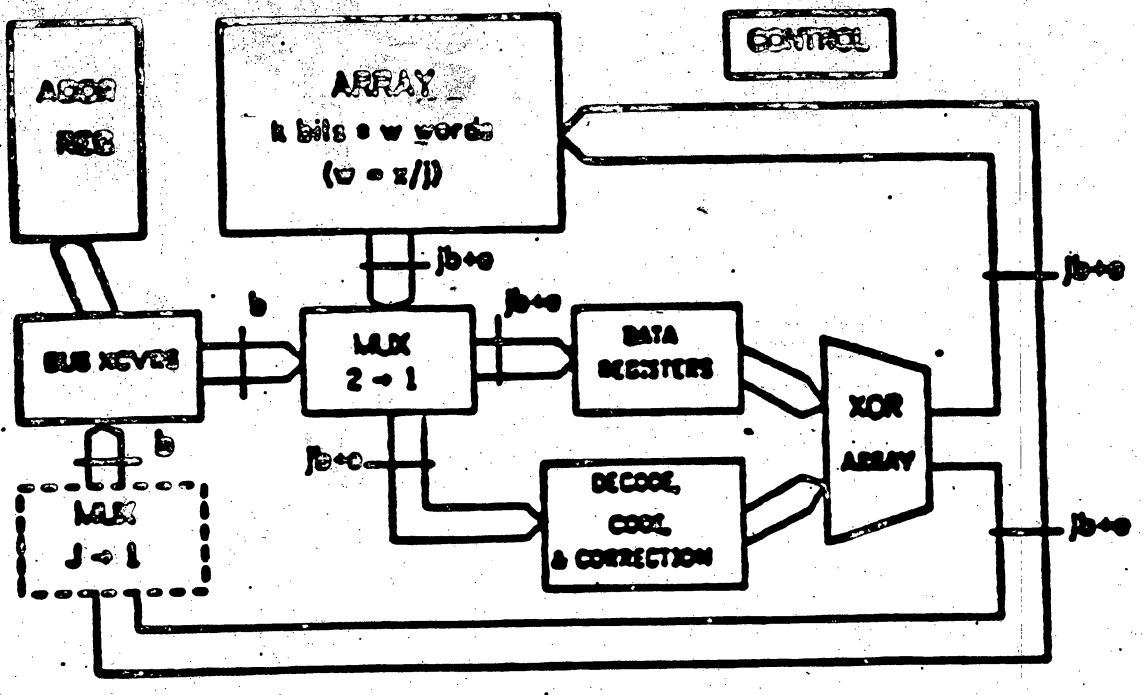




Nonredundant Memory Model

b bit words

x words



Hamming-Coded Memory Model

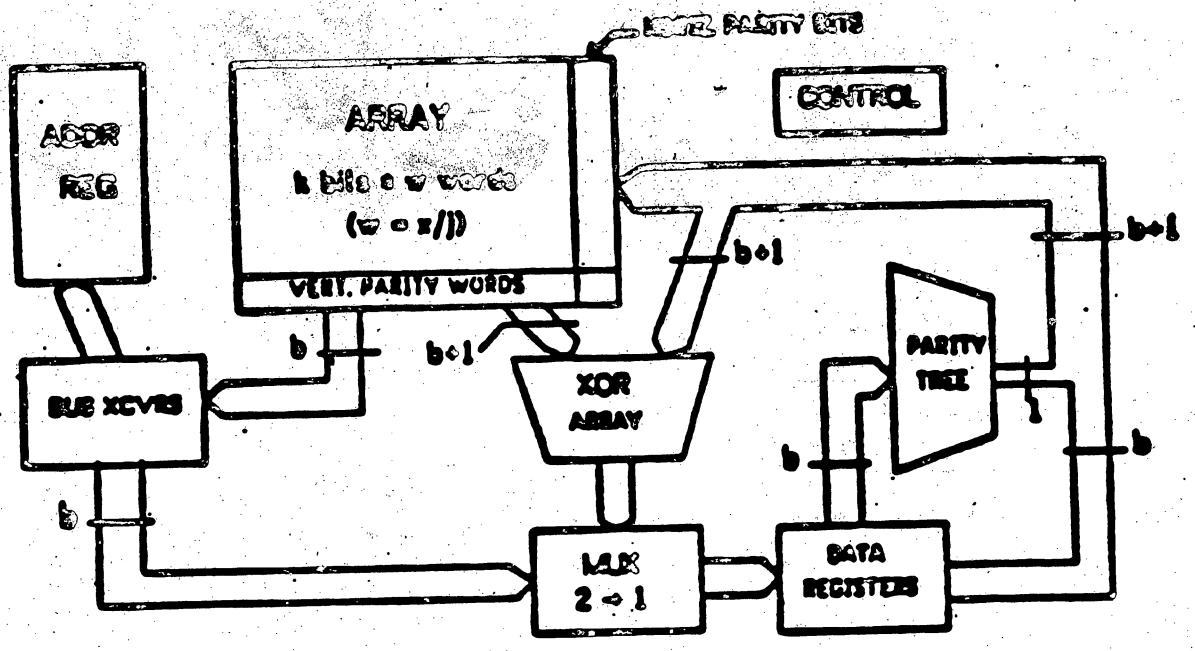
b bit logical words

x logical words

j logical words/physical word

e check bits/physical word

(x/j) physical words



Block-Coded Memory Model

b bit logical word

x logical words

b + 1 bit physical word

(x/j) additional physical words

MEMORY CHIP FAILURE MODES

%	4K MOS RAMs	PROMs
Whole Chip	0.0/11.8	17.9/0.0
Single Bit	92.0/35.3	53.9/100.0
Row(Column)	0.0/29.4	15.3/0.0
Unknown	8.0/23.5	12.9/0.0

Data indicates that whole chip failures are a small minority, rather than the dominant failure mode.

NONREDUNDANT MEMORY MODULE

exponential assumption

λ_k = failure rate for control circuit

λ_s = failure rate for selection circuit

λ_a = failure rate for memory array

$$R_{NR}(t) = e^{-(\lambda_k + \lambda_s + \lambda_a)t}$$

$$MTTF_{NR} = \frac{1}{\lambda_k + \lambda_s + \lambda_a}$$

$$Z_{NR}(t) = \lambda_k + \lambda_s + \lambda_a$$

SINGLE BIT FAILURE MODE

exponential assumption

λ_d = failure rate for decoding circuit

$\lambda_B = \lambda_d + \lambda_s + \lambda_k$

λ_b = memory bit failure rate

$$R_{SB}(t) = e^{-\lambda_e t}(ke^{-(k-1)\lambda_b t} - (k-1)e^{-k\lambda_b t})^w$$

k bits/word

w words

$$MTTF_{SB} = \int_0^\infty R(t)dt$$

$$= \frac{1}{\lambda_b}\left[\frac{1}{f_0} + \sum_{i=1}^{w} \frac{K_i \prod_{j=0}^{i-1} g_j}{\prod_{j=0} f_j}\right]$$

$$f_j = \omega k + \frac{\lambda_e}{\lambda_b} - j$$

$$g_j = \omega - j$$

$$Z_{SB}(t) = \lambda_e + \omega k(k-1)\lambda_b \frac{(1-e^{-\lambda_b t})}{(k-(k-1)e^{-\lambda_b t})}$$

Whole Chip Failure Mode

λ_c = memory chip failure mode

use d-bit chips such that no (code) word has more than one bit/chip

$\Rightarrow h = w/d$

$$R_{wc}(t) = e^{-\lambda_e t}(ke^{-(k-1)\lambda_c t} - (k-1)e^{-k\lambda_c t})^h$$

$$MTTF_{wc} = \frac{1}{\lambda_c}\left[\frac{1}{f_0} + \sum_{i=1}^{h} \frac{k^i \prod_{j=0}^{i-1} g_j}{\prod_{j=0}^{i} f_j}\right]$$

$$f_j = hk + \frac{\lambda_e}{\lambda_c} - j$$

$$g_j = \omega - j$$

$$Z_{wc} = \lambda_e + \frac{hk(k-1)\lambda_c(1-e^{-\lambda_c t})}{k-(k-1)e^{-\lambda_c t}}$$

ROW (COLUMN) FAILURE MODE

λ_r = memory row failure rate

use d-bit chips having q bits/row

$$\Rightarrow p = \frac{\omega q}{d}$$

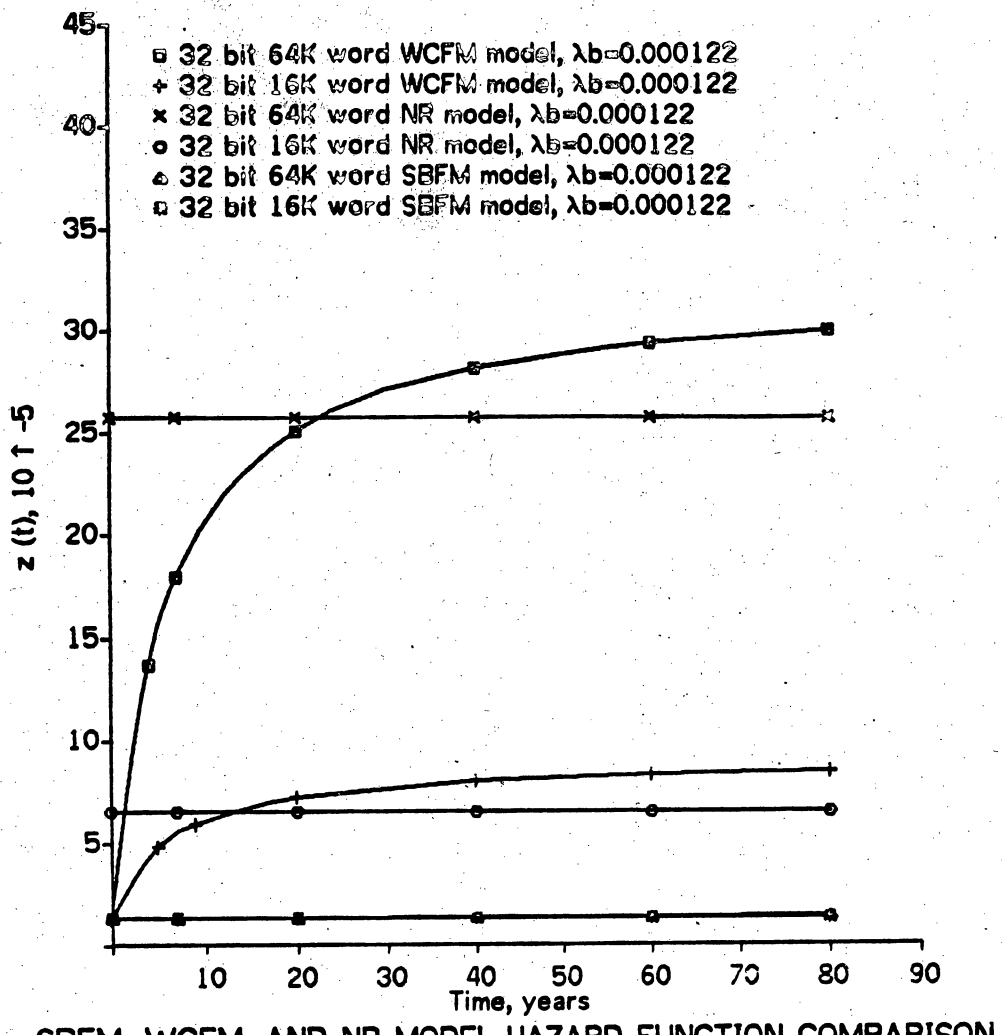
$$R_R(t) = e^{-\lambda_e t}(ke^{-(k-1)\lambda_r t} - (k-1)e^{-k\lambda_r t})^p$$

$$MTTF_R = \frac{1}{\lambda_r}\left[\frac{1}{f_0} + \sum_{i=1}^{p} \frac{k^i \prod_{j=0}^{i-1} g_j}{\prod_{j=0}^{i} f_j}\right]$$

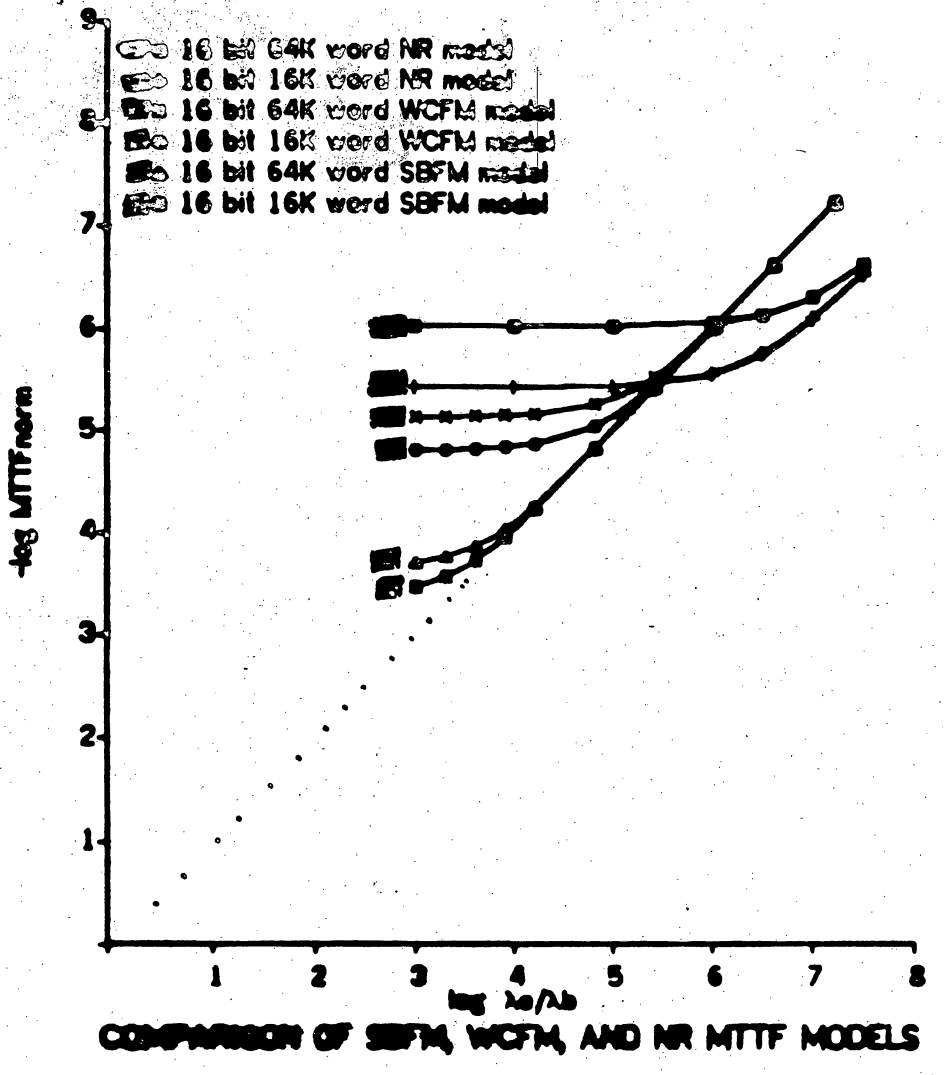
$$f_j = pk\frac{\lambda_e}{\lambda_r} - j$$

$$g_j = p - j$$

$$Z_R(t) = \lambda_e + \frac{pk(k-1)\lambda_r(1-e^{-\lambda_r t})}{k-(k-1)e^{-\lambda_r t}}$$

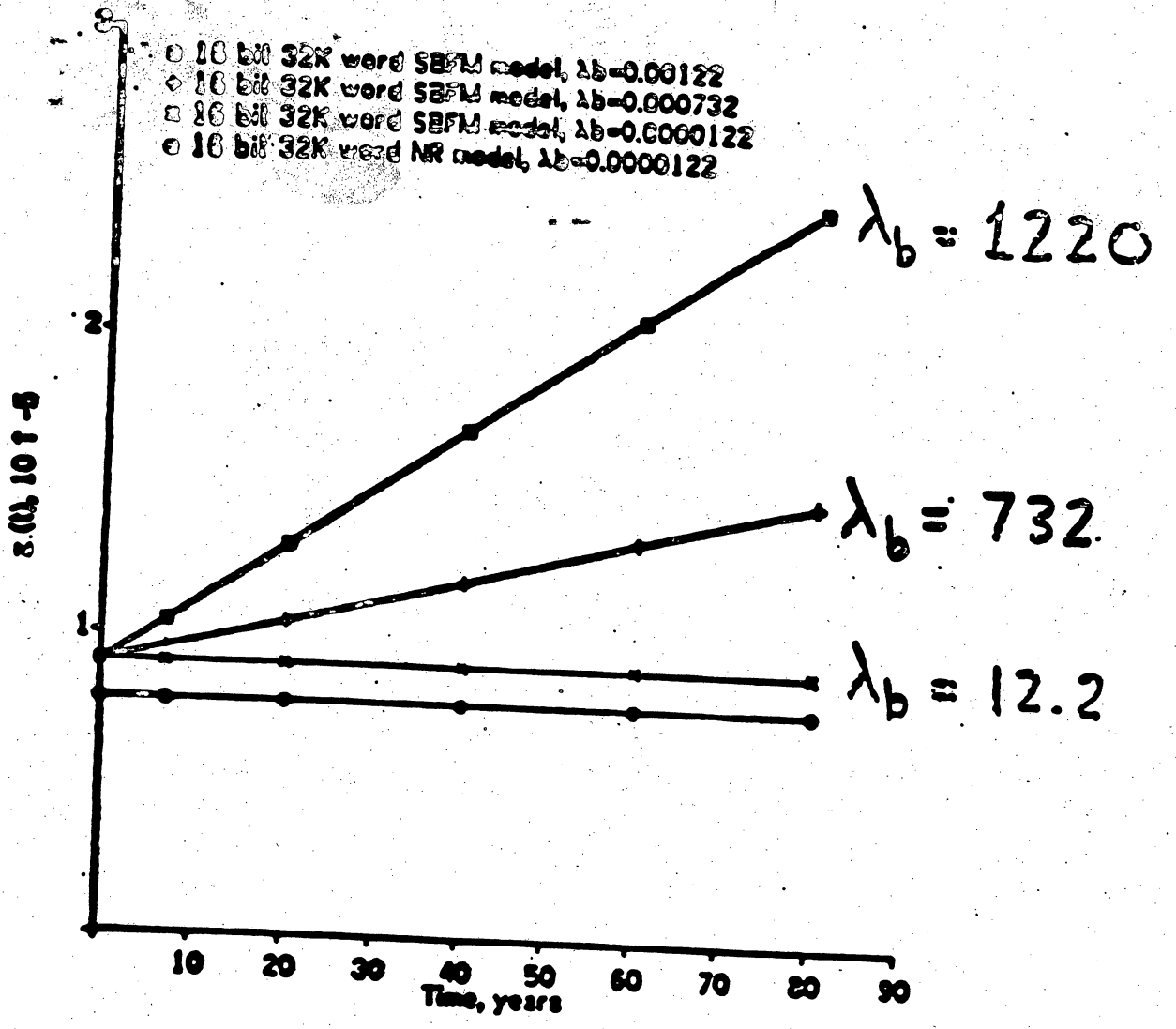


SBFM, WCFM, AND NR MODEL HAZARD FUNCTION COMPARISON

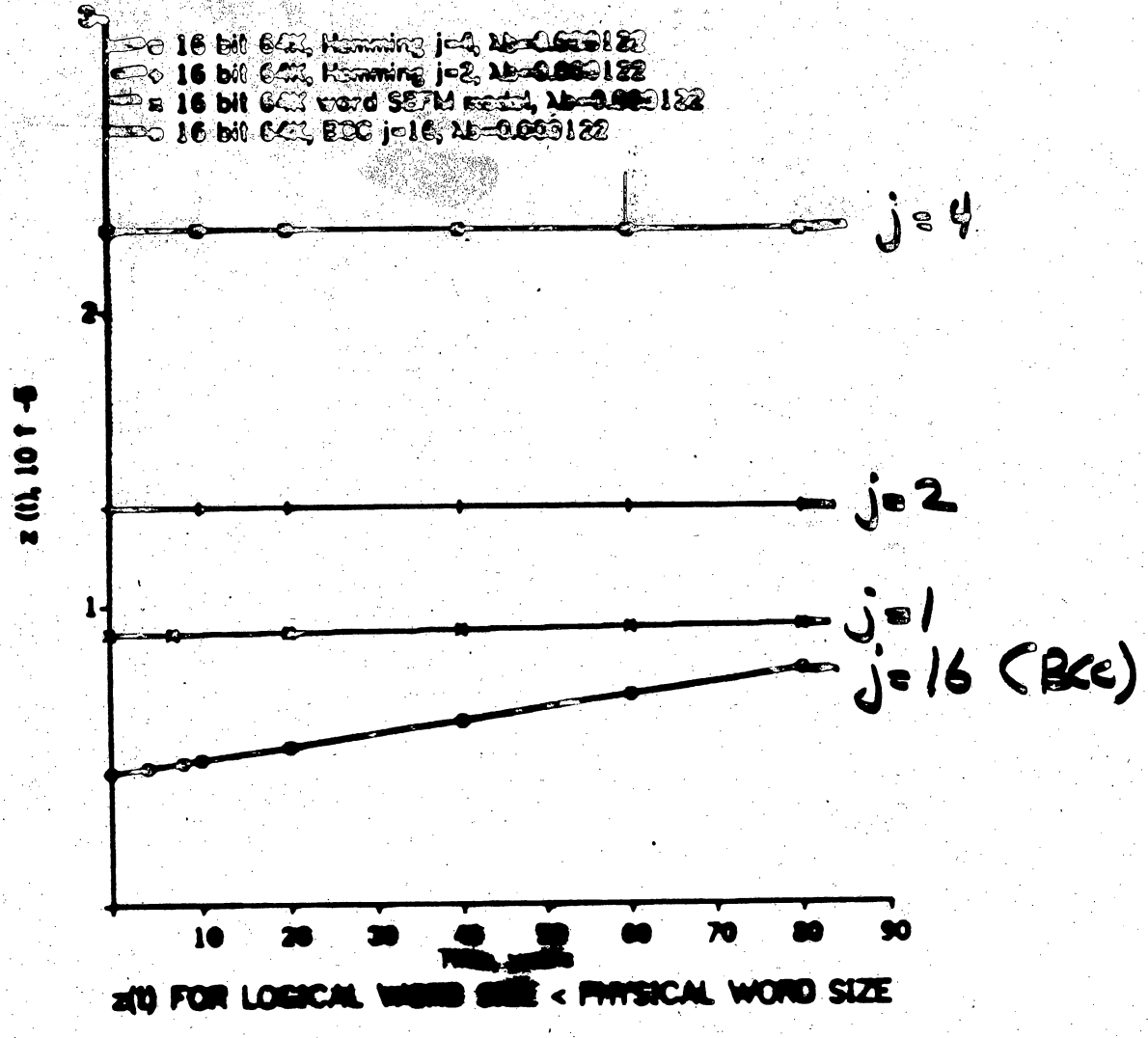


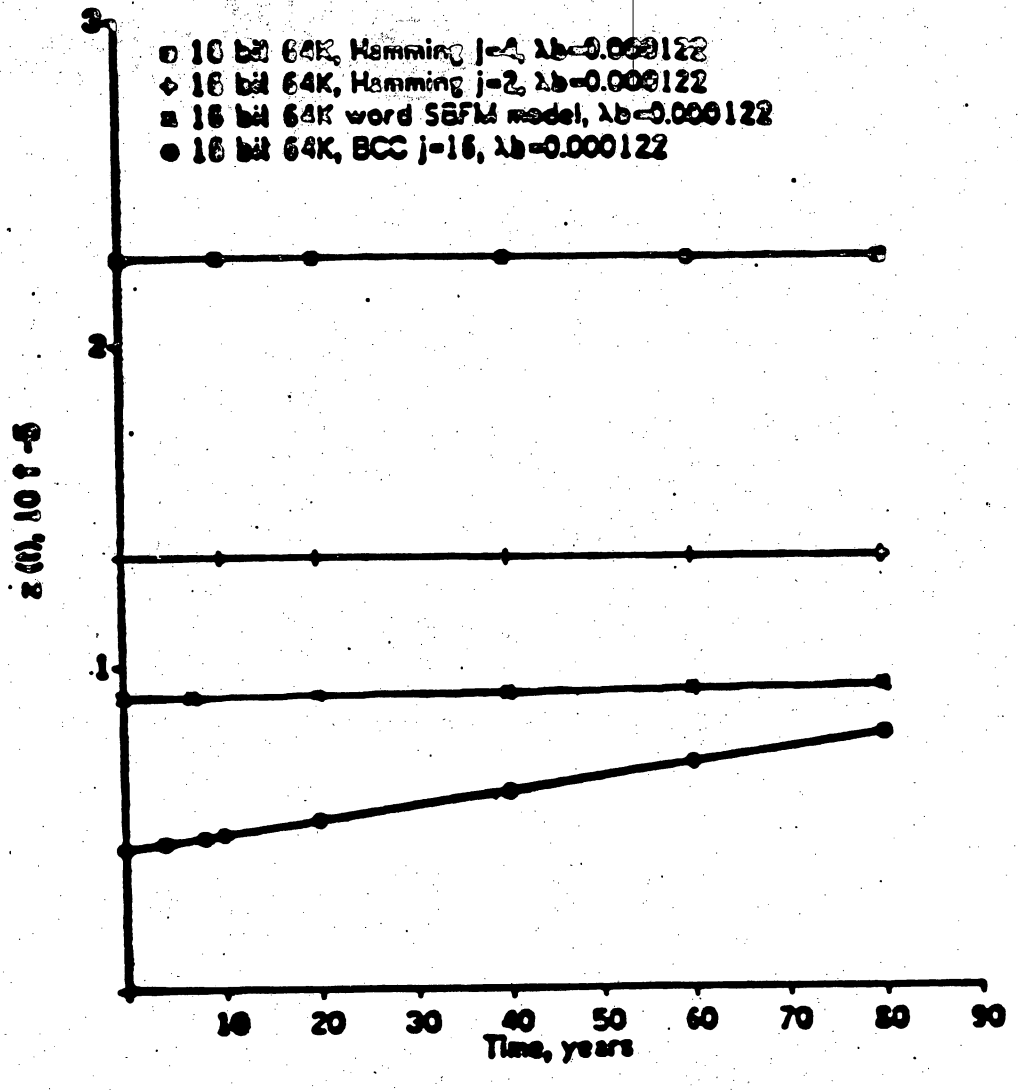
COMPARISON OF SBFM, WCFM, AND NR MTTF MODELS

MTTF normalized with respect to technology (i.e. as a function of (λ_o/λ_b) only) showing effect of (λ_o/λ_b) ratio on MTTF

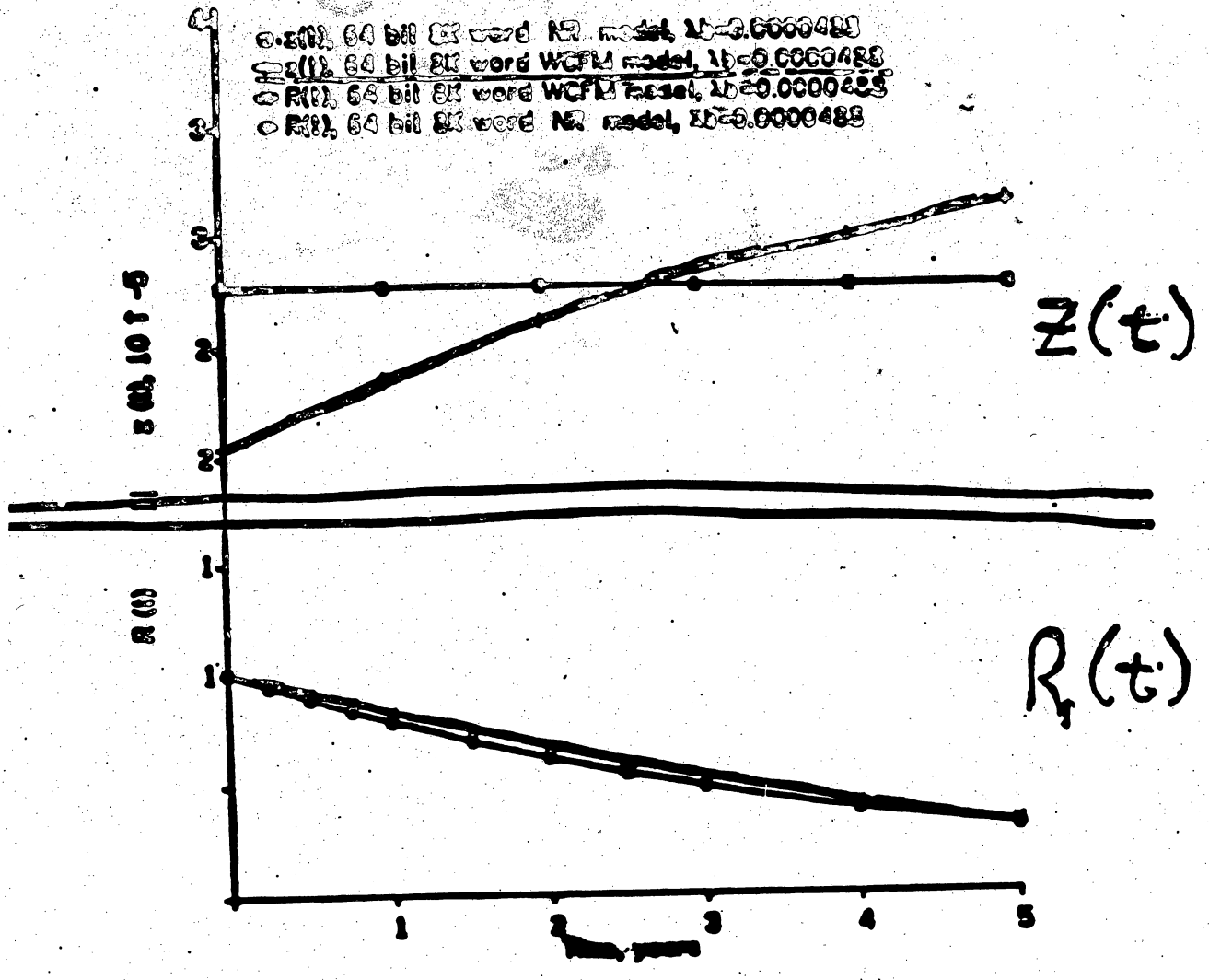


Sensitivity of the Hazard Function to λ_b


16 bit BCC, Hamming j=4,
16 bit BCC, Hamming j=2,
16 bit BCC word SEDM code,
16 bit BCC, BCC j=16,
j=4
j=2
j=1
j=16 (BCC)
3
2
1
10
20
30
70
80
90
z(1) FOR LOGICAL WORD SIZE < PHYSICAL WORD SIZE



Hazard Function for Logical Word Size Less than Physical Word Size



Choosing a 64 x 8K Memory

ECC vs. Nonredundant

Summary of Reliability Modeling

- Memory chip contribution to unreliability is negligible over a wide range of word size and memory size.

$\Rightarrow$ Control circuitry dominates unreliability

Performance in presence of failures

- Main memory expected degradation

  Call accessing patterns, Hamming and Block Code)

$$\frac{n\epsilon}{\omega}$$

n = number of failures

ω = number of words in memory

ϵ = fraction of a memory cycle degradation for correction

Microstore Expected Degradation

$$\frac{n\varepsilon}{w}$$

PROBABILITY DISTRIBUTION OF PERFORMANCE DEGRADATION

(HAMMING CODE)

(PDP-11 CLASS MACHINE)

DEGRADATION	PROBABILITY
$< .001$	.657
$< .01$	.269
$< .02$	.023
$< .06$	.051

PROBABILITY DISTRIBUTION OF PERFORMANCE DEGRADATION

(BCC)

(VAX-11/780 CLASS MACHINE)

DEGRADATION	PROBABILITY
< .001	.048
< .01	.607
< .02	.167
< .05	.036
< .1	.083
< .2	.024
< .7	.035

Conclusions

- memory chip failure modes affect system reliability and dictate the choice of models for reliability
- these models provide good tools for design tradeoff analyses
- ECC memories are not automatically more reliable than NR memories, since their reliability is strongly influenced by the reliability of the control circuitry, and possibly dominated by it
    - a strong argument for LSI control circuitry
- ECC memories experience negligible performance degradation with errors present, in most cases
- Block coded memories, while more reliable than Hamming coded memories, experience slightly more degradation in the presence of errors, although the degradation is usually negligible

Reconfiguration

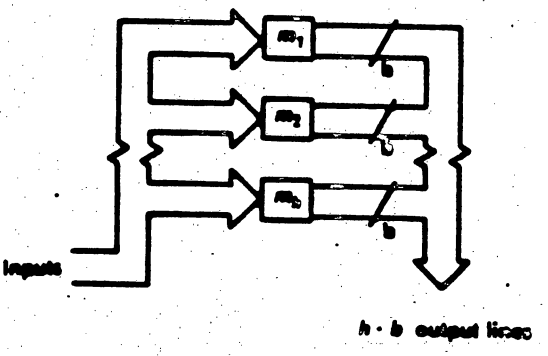
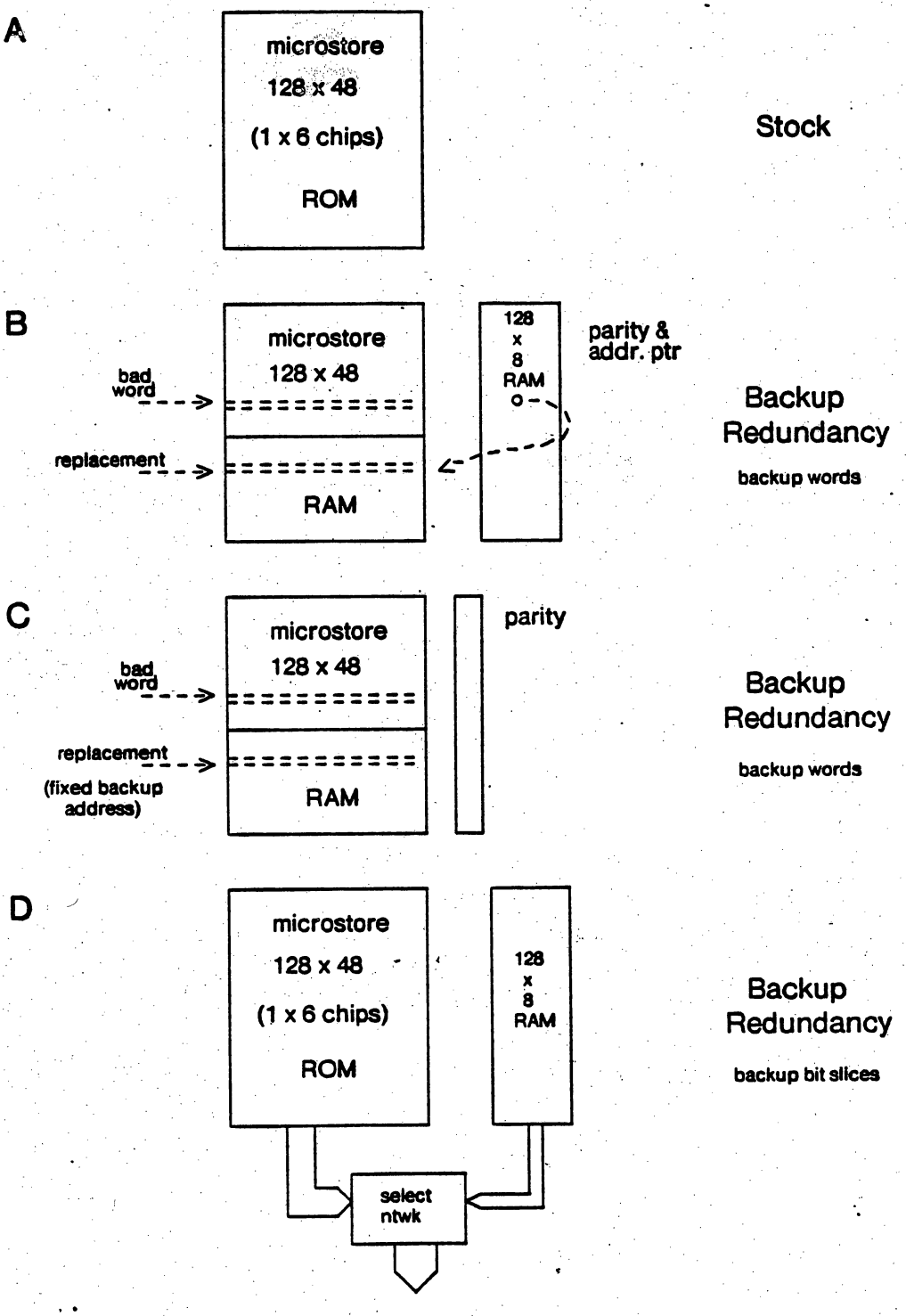


Figure 3–78. Circuit design of parallel byte-wide modules.


A
microstore
128 x 48
(1 x 6 chips)
ROM
Stock
B
microstore
128 x 48
bad word
replacement
RAM
128 x 8 RAM
parity & addr. ptr
Backup Redundancy
backup words
C
microstore
128 x 48
bad word
replacement
(fixed backup address)
RAM
parity
Backup Redundancy
backup words
D
microstore
128 x 48
(1 x 6 chips)
ROM
128 x 8 RAM
select ntwk
Backup Redundancy
backup bit slices

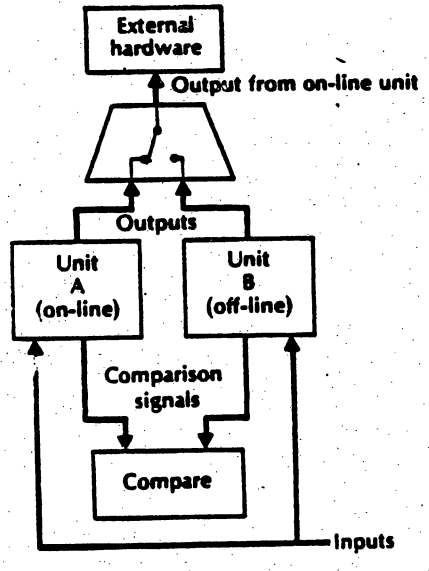


Figure 3–69. Reconfigurable duplication. A detected mismatch during comparison of characteristic signals triggers reconfiguration.

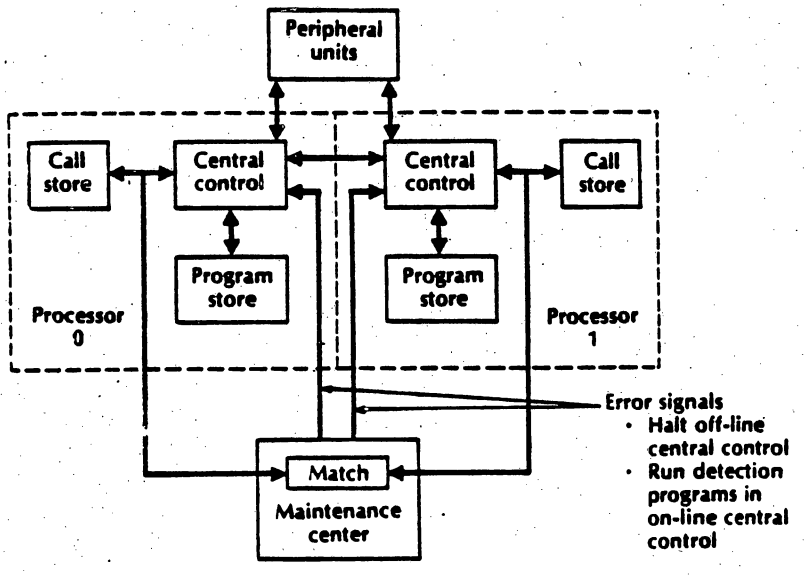
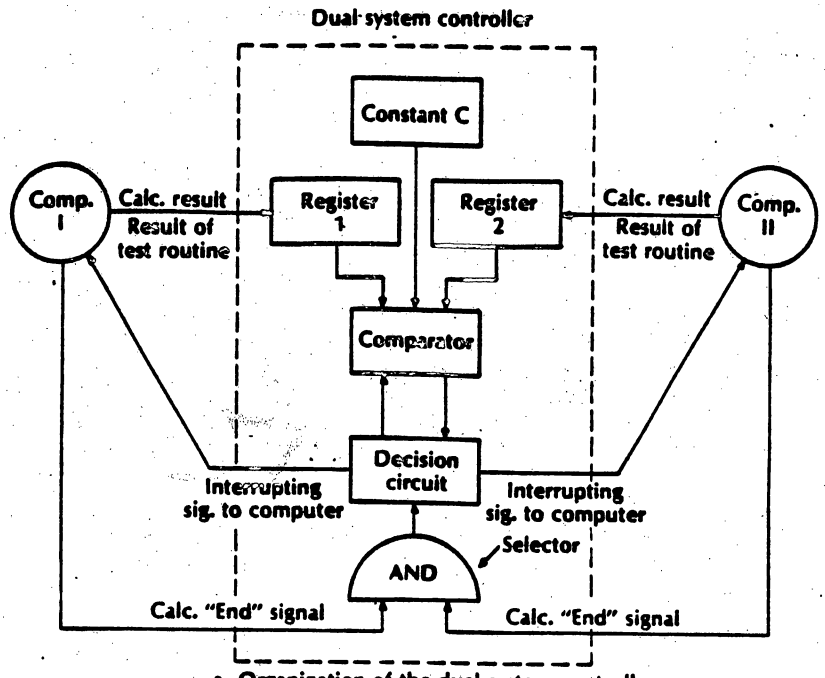
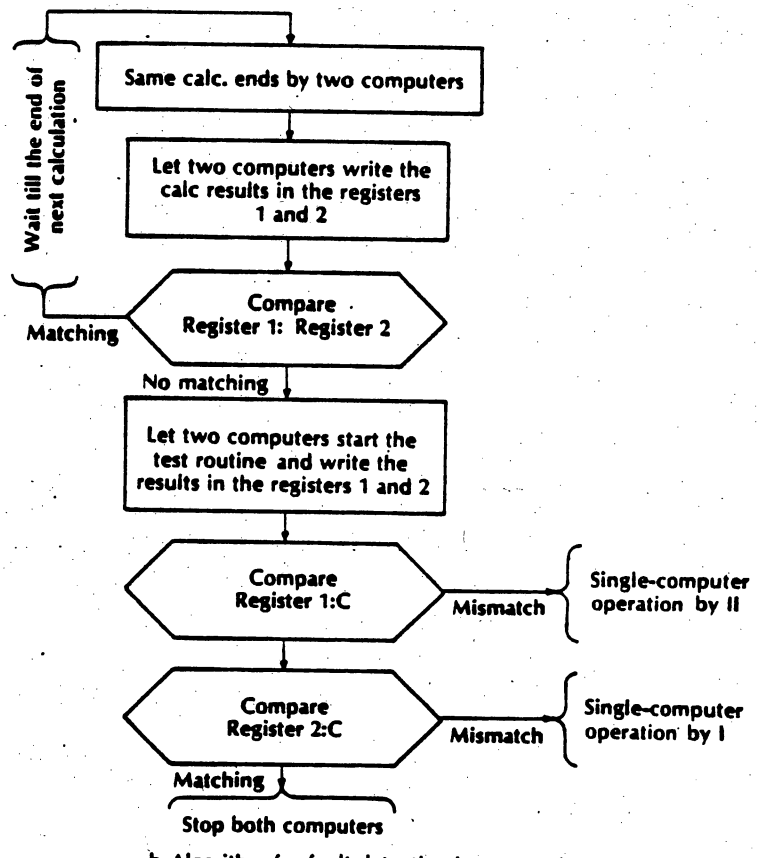


Figure 3–70. Reconfigurable duplication in Bell ESS-2. The two processors run synchronously; comparison of the call store input registers is performed constantly.



a. Organization of the dual system controller with respect to configuration control

Figure 3–71. Synchronization, matching, and reconfiguration in the COM-TRAC computer. Synchronization and matching are performed at the task level, [Ihara et al., 1978]. (© IEEE 1978.)



b. Algorithm for fault detection by comparison, synchronization, and reconfiguration

Figure 3–71—*Continued*

Formulation of Markov Models

Every system has a set of opertional and failed "states", each of which represents a particular combination of working and failed modules.

=> For a system with N modules, the basic model has 2^N states.

Markov Assumptions

1. The probability of a given state transition depends only on the current state.
2. The length of time spent in a state does not influence either the probability distribution of the remaining time until the next transition or the probability of the next state transition path.

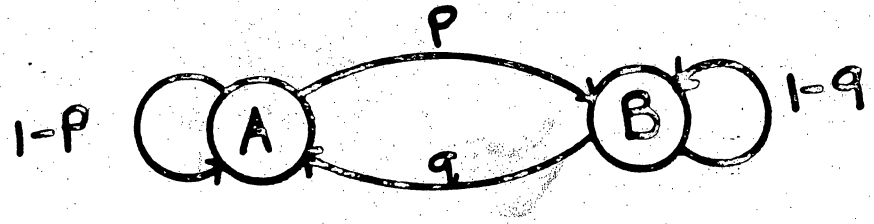
DIRECT RESULTS:

DISCRETE TIME WAITING TIMES ARE GEOMETRICALLY DISTRIBUTED

CONTINUOUS TIME WAITING TIMES ARE EXPONENTIALLY DISTRIBUTED

Discrete Time Markov Model

directed graph



A,B - states

p,q - transition probabilities

transition probability matrix

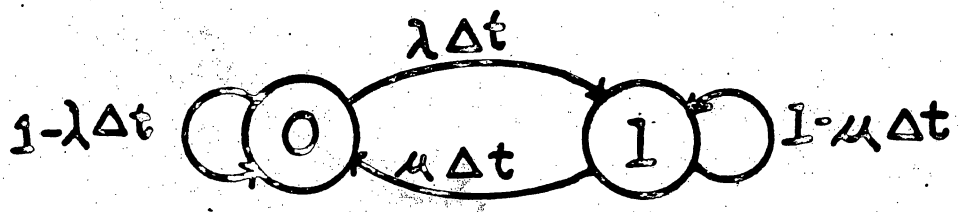
$$\begin{array}{c} \\ A \\ B \end{array} \begin{array}{c} \begin{array}{cc} A & B \end{array} \\ \begin{bmatrix} 1-p & p \\ q & 1-q \end{bmatrix} \end{array} = \Psi$$

$$[P_A^{k+1}, P_B^{k+1}] = [P_A^{k}, P_B^{k}] \times \Psi$$

$$\vec{P}(k+1) = \vec{P}(k) \times \Psi$$

Derivation of the Continuous Markov Model

let $p = \lambda \Delta t$, $q = \mu \Delta t$



$$\vec{P}(t+\Delta t) = \vec{P}(t) \times \Psi$$

$$\vec{P}(t+\Delta t) = [P_0(t), P_1(t)] \times \begin{bmatrix} 1-\lambda\Delta t & \lambda\Delta t \\ \mu\Delta t & 1-\mu\Delta t \end{bmatrix}$$

$$P_0(t+\Delta t) = P_0(t)(1-\lambda\Delta t) + P_1(t)\,\mu\Delta t$$

$$P_0(t+\Delta t) = P_0(t) - \lambda P_0(t)\Delta t + \mu P_1$$

$$\frac{P_0(t+\Delta t) - P_0(t)}{\Delta t} = -\lambda P_0(t) + \mu P_1(t)$$

take limit as $\Delta t \longrightarrow 0$

$$\frac{dP_0(t)}{dt} = -\lambda P_0(t) + \mu P_1(t)$$

Derivation cont.

Similarly,

$$P_1(t+\Delta) = (1-\mu\Delta t)P_1(t) + \lambda\Delta t P_0(t)$$

$$\frac{dP_1(t)}{dt} = \lambda P_0(t) - \mu P_1(t)$$

$$\dot{P}_0(t) = -\lambda P_0(t) + \mu P_1(t)$$

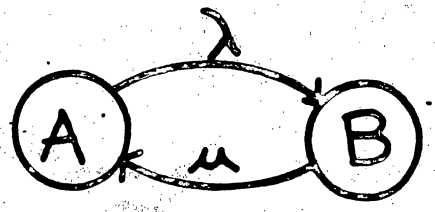
$$\dot{P}_1(t) = \lambda P_0(t) - \mu P_1(t)$$

$$\boxed{\dot{\vec{P}}(t) = \vec{P}(t) \times T}$$

$$T = \begin{bmatrix} -\lambda & \lambda \\ \mu & -\mu \end{bmatrix}$$

Continuous Time Markov Model

directed graph



A,B - states

λ, μ - transition rates

transition probability matrix

$$\begin{array}{c} \\ A \\ B \end{array}\begin{array}{c} \begin{array}{cc} A & B \end{array} \\ \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \end{array} = \Psi$$

waiting time parameter matrix

$$\begin{array}{c} \\ A \\ B \end{array}\begin{array}{c} \begin{array}{cc} A & B \end{array} \\ \begin{bmatrix} \lambda & 0 \\ 0 & \mu \end{bmatrix} \end{array} = \Lambda$$

$$\boxed{T = \Lambda(\Psi - I)} = \begin{bmatrix} -\lambda & \lambda \\ \mu & -\mu \end{bmatrix}$$

$$= \begin{bmatrix} \lambda & 0 \\ 0 & \mu \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}$$

Analytic Solution Technique

$$\vec{P}(t) = \vec{P}(t) \times T$$

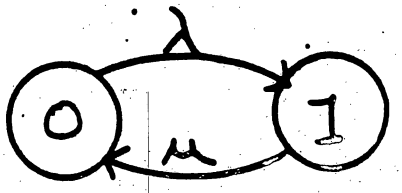
THIS IS A SET OF LINEAR CONSTANT COEFFICIENT DIFFERENTIAL EQUATIONS. LAPLACE TRANSFORMS ARE USEFUL FOR SOLVING SUCH SYSTEMS OF EQUATIONS BECAUSE THEY CHANGE THE PROBLEM INTO ONE OF SOLVING A SET OF LINEAR ALGEBRAIC EQUATIONS, WHICH IS (RELATIVELY) EASY.

$$\mathcal{L}\{f(t)\} = F(s) = \int_0^\infty f(t)e^{-st}\,dt$$

LaPlace Transform Table

f(t)	F(s)
k	$\frac{k}{s}$
e^{-at}	$\frac{1}{s+a}$
$\dot{f}(t)$	$sF(s) - f(0)$
$tf(t)$	$-\dot{F}(s)$
$\int_0^t f(t) = t$	$\frac{1}{s}F(s)$
$\frac{1}{t}f(t)$	$\int_0^{\infty} F(x)\,dx$
$f_1(t) + f_2(t)$	$F_1(s) + F_2(s)$

2-State Markov Model



$$T = \begin{bmatrix} -\lambda & \lambda \\ \mu & -\mu \end{bmatrix}$$ as developed earlier

$$\mathcal{L}\{\dot{P}_0(t)\} = \mathcal{L}\{-\lambda P_0(t) + \mu P_1(t)\}$$

$$\mathcal{L}\{\dot{P}_1(t)\} = \mathcal{L}\{\lambda P_0(t) - \mu P_1(t)\}$$

$$s p_0(s) - P_0(0) = -\lambda p_0(s) + \mu p_1(s) \qquad (1a)$$

$$s p_1(s) - P_1(0) = \lambda p_0(s) - \mu p_1(s)$$

$$(s+\lambda) p_0(s) - \mu p_1(s) = P_0(0)$$

$$-\lambda p_0(s) + (s+\mu) p_1(s) = P_1(0)$$

$$\begin{bmatrix} p_0(s), p_1(s) \end{bmatrix} \times \begin{bmatrix} s+\lambda & -\lambda \\ -\mu & s+\mu \end{bmatrix} = \begin{bmatrix} P_0(0), P_1(0) \end{bmatrix}$$

2-State Markov Model cont.

$$\vec{\rho}(s) \times A = \vec{P}(0)$$

with $$A = [sI - T]$$

desired set of equations is

$$\rho(s) = P(0) \times A^{-1}$$

to calculate A^{-1} with elements α_{ij},

$$\alpha_{ij} = \frac{\text{cofactor } ji(A)}{\det A}$$

det A = determinant of A

cofactor ji(A) = $(-1)^{j+i}$ x determinant of matrix formed by removing row j and column i from A

2-State Markov Model cont.

for the 2-state system,

$$A = \begin{bmatrix} s+\lambda & -\lambda \\ -\mu & s+\mu \end{bmatrix}$$

$$\det A = (s+\lambda)(s+\mu) - \lambda\mu$$

$$= s(s+\lambda+\mu)$$

$$A^{-1} = \frac{1}{s(s+\lambda+\mu)} \begin{bmatrix} s+\mu & \lambda \\ \mu & s+\lambda \end{bmatrix}$$

assuming the system was operational (state 0) at time $t = 0$,

$$\vec{P}(0) = [1\ 0]$$

so

$$\vec{P}(s) = [1\ 0] \begin{bmatrix} \frac{s+\mu}{s(s+\lambda+\mu)} & \frac{\lambda}{s(s+\lambda+\mu)} \\ \frac{\mu}{s(s+\lambda+\mu)} & \frac{s+\lambda}{s(s+\lambda+\mu)} \end{bmatrix}$$

2-State Markov Model cont.

$$P_0(s) = \frac{s+\mu}{s(s+\lambda+\mu)}$$

$$= \frac{1}{s+\lambda+\mu} + \frac{\mu}{s(s+\lambda+\mu)}$$

using partial fraction expansion,

$$P_0(s) = \frac{1}{s+\lambda+\mu} + \frac{A}{s} + \frac{B}{s+\lambda+\mu}$$

$$A(s+\lambda+\mu) + Bs = \mu$$

$$(A+B)s + A(\lambda+\mu) = \mu$$

$$\Rightarrow A+B=0$$

$$A = \frac{\mu}{\lambda+\mu}, \quad B = -\frac{\mu}{\lambda+\mu}$$

$$P_0(s) = \frac{1-\frac{\mu}{\lambda+\mu}}{s+\lambda+\mu} + \frac{\frac{\mu}{\lambda+\mu}}{s}$$

$$\boxed{P_0(s) = \frac{\frac{\mu}{\lambda+\mu}}{s} + \frac{\frac{\lambda}{\lambda+\mu}}{s+\lambda+\mu}}$$

2-State Markov Model cont.

$$p_I(s) = \frac{\lambda}{s(s+\lambda+\mu)}$$

as before,

$$p_1(s) = \frac{A}{s} + \frac{B}{s+\lambda+\mu}$$

$$A(s+\lambda+\mu) + Bs = \lambda$$

$$(A+B)s + A(\lambda+\mu) = \lambda$$

$$\Rightarrow A+B=0$$

$$A = \frac{\lambda}{\lambda+\mu}$$

$$B = -\frac{\lambda}{\lambda+\mu}$$

$$\boxed{p_1(s) = \frac{\frac{\lambda}{\lambda+\mu}}{s} - \frac{\frac{\lambda}{\lambda+\mu}}{s+\lambda+\mu}}$$

2-State Markov Model cont.

$$P_0(t) = \frac{\mu}{\lambda+\mu} + \frac{\lambda}{\lambda+\mu} e^{-(\lambda+\mu)t}$$

$$= P_r\{\text{operational at time } t\}$$

$$P_1(t) = \frac{\lambda}{\lambda+\mu} - \frac{\lambda}{\lambda+\mu} e^{-(\lambda+\mu)t}$$

$$= P_r\{\text{failed at time } t\}$$

define Availability = A(t) = $P_0(t)$

steady state availability = $\frac{\mu}{\lambda+\mu}$

note that $MTTF = \frac{1}{\lambda}$

$$MTTR = \frac{1}{\mu}$$

some algebra: $\frac{\mu}{\lambda+\mu} = \frac{\frac{\mu}{\mu\lambda}}{\frac{\lambda}{\mu\lambda}+\frac{\mu}{\mu\lambda}}$

$$= \frac{\frac{1}{\lambda}}{\frac{1}{\mu}+\frac{1}{\lambda}}$$

steady state availability $= \frac{MTTF}{MTTR + MTTF}$

Steady State Calculations

to calculate only the steady state values, redo equation (1):

$\dot{P}_0(t)$ and $\dot{P}_1(t) \rightarrow 0$

$P_0(t)$ and $P_1(t) \rightarrow P_0$ and P_1

$$0 = -\lambda P_0 + \mu P_1$$
$$0 = \lambda P_0 - \mu P_1$$

remember - $P_0 + P_1 = 1$

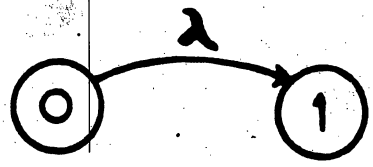
$$P_0 = 1 - P_1$$
$$P_0 = 1 - \frac{\lambda}{\mu} P_0$$
$$P_0 \left(1 + \frac{\lambda}{\mu}\right) = 1$$
$$P_0 \left(\frac{\mu + \lambda}{\mu}\right) = 1$$

$$\boxed{P_0 = \left(\frac{\mu}{\lambda + \mu}\right)}$$

$$\Rightarrow \boxed{P_1 = \frac{\lambda}{\lambda + \mu}}$$

Reliability of 2-State Model

In order to calculate the reliability rather than the availability, the failed state is made into a "trapping" state.



$$\Psi = \begin{bmatrix} 0 & 1 \end{bmatrix} \qquad \Lambda = \begin{bmatrix} \lambda & 0 \\ 0 & 0 \end{bmatrix}$$

$$T = \begin{bmatrix} \lambda & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} -\lambda & \lambda \\ 0 & 0 \end{bmatrix}$$

applying $\dot{\vec{P}}(t) = \vec{P}(t) \times T$

$$\begin{cases} \dot{P}_0(t) = -\lambda P_0(t) \\ \dot{P}_1(t) = \lambda P_0(t) \end{cases}$$

$$A = [sI - T] = \begin{bmatrix} s+\lambda & \lambda \\ 0 & s \end{bmatrix}$$

Reliability of 2-State Model cont.

$$A^{-1} = \frac{1}{s(s+\lambda)} \begin{bmatrix} s & \lambda \\ 0 & s+\lambda \end{bmatrix}$$

$$\vec{P}(0) = [1\ 0] \begin{bmatrix} \frac{s}{s(s+\lambda)} & \frac{\lambda}{s(s+\lambda)} \\ 0 & \frac{s+\lambda}{s(s+\lambda)} \end{bmatrix}$$

$$\pi_0(s) = \frac{s}{s(s+\lambda)} = \frac{1}{s+\lambda}$$

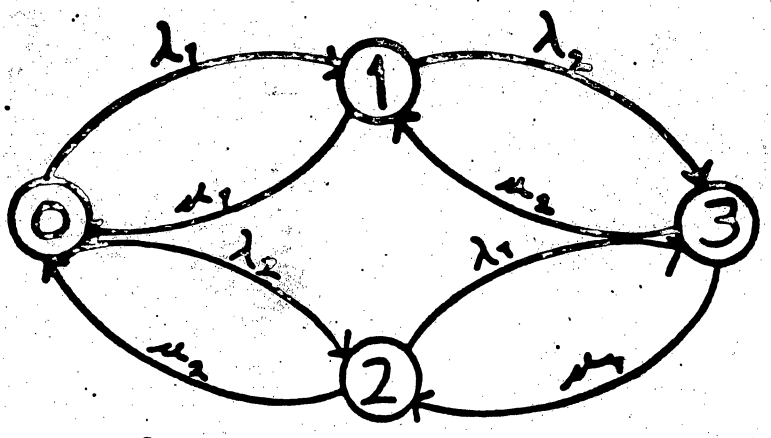
$$\pi_1(s) = \frac{\lambda}{s(s+\lambda)} = \frac{1}{s} - \frac{1}{s+\lambda}$$ by partial fraction expansion

taking the inverse transform,

$$P_0(t) = e^{-\lambda t} = R(t) - \text{as expected}$$

$$P_1(t) = 1 - e^{-\lambda t}$$

Example: Dual Processor System



λ_i = failure rate of processor i

μ_i = repair rate of processor i

state 0: both operational

state 1: processor 1 failed

state 2: processor 2 failed

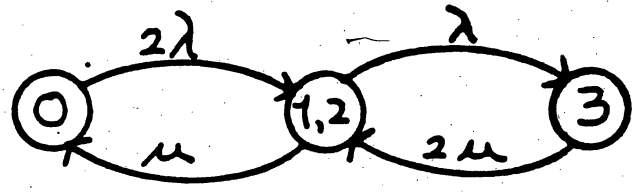
state 3: both failed

Dual Processor cont.

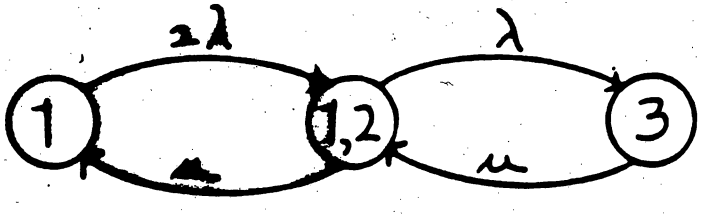
often, assume $\lambda_1 = \lambda_2 = \lambda$

$\mu_1 = \mu_2 = \mu$

then the previous model can be collapsed to



NOTE THAT THIS MODEL ASSUMES AS MANY REPAIRMEN AS FAILED MACHINES. IF ONLY ONE REPAIRMAN IS AVAILABLE, THE MODEL BECOMES



Dual Processor cont.

by inspection from the preceding diagram,

$$\Psi = \begin{bmatrix} 0 & 1 & 0 \\ \frac{\mu}{\lambda+\mu} & 0 & \frac{\lambda}{\lambda+\mu} \\ 0 & 1 & 0 \end{bmatrix} \qquad \Lambda = \begin{bmatrix} 2\lambda & 0 & 0 \\ 0 & \lambda+\mu & 0 \\ 0 & 0 & \mu \end{bmatrix}$$

$$\Rightarrow \Gamma = \begin{bmatrix} -2\lambda & 2\lambda & 0 \\ \mu & -\lambda-\mu & \lambda \\ 0 & \mu & -\mu \end{bmatrix} \qquad \text{Note } \Gamma = \Lambda(\Psi - I)$$

$$A = \begin{bmatrix} s+2\lambda & -2\lambda & 0 \\ -\mu & s+\lambda+\mu & -\lambda \\ 0 & -\mu & s+\mu \end{bmatrix}$$

need to solve

$$\vec{\rho}(s) = \vec{P}(0) \times A^{-1}$$

assume $\vec{P}(0) = [1\ 0\ 0]$

Dual Processor cont.

$$\det A = (s+2\lambda)(s+\lambda+\mu)(s+\mu) - 2\lambda\mu(s+\mu) - \lambda\mu(s+2\lambda$$

$$= s(s+r_1)(s+r_2)$$

$$r_1 = \frac{1}{2}(3\lambda+2\mu+\sqrt{\lambda^2+4\lambda\mu})$$

$$r_2 = \frac{1}{2}(3\lambda+2\mu-\sqrt{\lambda^2+4\lambda\mu})$$

$$A^{-1} = \frac{1}{\det A}\begin{bmatrix} \det\begin{bmatrix} s+\lambda+\mu & -\lambda \\ -\mu & s+\mu \end{bmatrix} & -\det\begin{bmatrix} -2\lambda & 0 \\ -\mu & s+\mu \end{bmatrix} & \cdots \\ -\det\begin{bmatrix} -\mu & -\lambda \\ 0 & s+\mu \end{bmatrix} & \det\begin{bmatrix} s+2\lambda & 0 \\ 0 & s+\mu \end{bmatrix} & \cdots \\ \det\begin{bmatrix} \mu & s+\lambda+\mu \\ 0 & -\mu \end{bmatrix} & -\det\begin{bmatrix} s+2\lambda & -2\lambda \\ 0 & -\mu \end{bmatrix} & \cdots \end{bmatrix}$$

$$A^{-1} = \begin{bmatrix} \frac{s^2+(\lambda+2\mu)s+\mu^2}{s(s+r_1)(s+r_2)} & \frac{2\lambda s+2\lambda\mu}{s(s+r_1)(s+r_2)} & \frac{2\lambda^2}{s(s+r_1)(s+r_2)} \\ \frac{\mu s+\mu^2}{s(s+r_1)(s+r_2)} & \frac{s^2+(2\lambda+\mu)s+2\lambda\mu}{s(s+r_1)(s+r_2)} & \frac{\lambda s+2\lambda^2}{s(s+r_1)(s+r_2)} \\ \frac{\mu^2}{s(s+r_1)(s+r_2)} & \frac{\mu s+2\lambda\mu}{s(s+r_1)(s+r_2)} & \frac{s^2+(3\lambda+\mu)s+2\lambda^2}{s(s+r_1)(s+r_2)} \end{bmatrix}$$

Dual Processor cont.

$$P_3(s) = \frac{2\lambda^2}{s(s+r_1)(s+r_2)}$$

$$= \frac{A}{s} + \frac{B}{s+r_1} + \frac{C}{s+r_2}$$

$$2\lambda^2 = A(s^2+(r_1+r_2)s+r_1 r_2)+B(s^2+r_2 s)+C(s^2+r_1 s)$$

$$\therefore \lambda^2 = (A+B+C)s^2+[(A+B)r_2+(A+C)r_1]s+Ar_1 r_2$$

$$\vdots$$

$$A = \frac{2\lambda^2}{r_1 r_2} = \frac{2\lambda^2}{2\lambda^2+2\lambda\mu+\mu}$$

$$B = A\frac{r_2}{r_1-r_2} = \frac{4\lambda^2}{\lambda^2+4\lambda\mu+(3\lambda+2\mu)\sqrt{\lambda^2+4\lambda\mu}}$$

$$C = A\frac{r_1}{r_2-r_1} = \frac{4\lambda^2}{\lambda^2+4\lambda\mu-(3\lambda+2\mu)\sqrt{\lambda^2+4\lambda\mu}}$$

taking the inverse transform,

$$P_3(t) = A + Be^{-r_1 t} + Ce^{-r_2 t}$$

Dual Processor cont.

after extensive manipulation,

$$P_3(t) = \frac{2\lambda^2}{2\lambda^2+2\lambda\mu+\mu^3}\left[1 - e^{-\frac{1}{2}(3\lambda+2\mu)t} \times \left(\cosh\left[\frac{1}{2}\sqrt{\lambda^2+4\lambda\mu}\,t\right] + \frac{3\lambda+2\mu}{\sqrt{\lambda^3+4\lambda\mu}}\sinh\left[\frac{1}{2}\sqrt{\lambda^2+4\lambda\mu}\,t\right]\right)\right]$$

let $Q = 3\lambda + 2\mu$

$$R = \sqrt{\lambda^2+4\lambda\mu}$$

NOTE: $P_3(t)$ is the "unavailability"

$$A(t) = 1 - P_3(t)$$

$$A(t) = 1 - \frac{2\lambda^2}{2\lambda^2+2\lambda\mu+\mu^2}\left[1 - e^{-\frac{1}{2}Qt}\left(\cosh\left[\frac{1}{2}Rt\right] + \frac{Q}{R}\sinh\left[\frac{1}{2}Rt\right]\right)\right]$$

steady state availability

$$= \lim_{t\to\infty} A(t) = \frac{2\lambda\mu+\mu^2}{2\lambda^2+2\lambda\mu+\mu^2}$$

IF all you're interested in is steady state, no average.

Combinatorial

$$A_{sys} = 1 - (1-A)^2$$
$$= 2A - A^2$$

Queuing Model

M/M/c/k/k

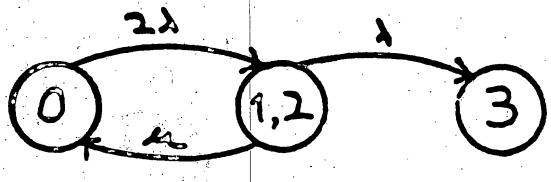
$$P_n = \left(\frac{k-n+1}{n}\right)(\lambda/\mu) P_{n-1}$$
$$n = 1, 2, \ldots c$$

$$P_n = \left(\frac{k-n+1}{c}\right)(\lambda/\mu) P_{n-1}$$

$$A_{sys} = P_0 + P_1$$
$$= \frac{\mu^2 + 2\lambda\mu}{\mu^2 + 2\lambda\mu + \lambda^2}$$
$$= \frac{\mu^2}{(\mu+\lambda)^2} + \frac{2\lambda\mu}{(\mu+\lambda)^2}$$
$$= A^2 + 2\frac{\mu}{\mu+\lambda}\left(\frac{\lambda}{\mu+\lambda}\right)$$
$$= A^2 + 2A(1-A)$$
$$= 2A - A^2$$

DUAL PROCESSOR RELIABILITY

To find the reliability, change the diagram to



$$\Psi = \begin{bmatrix} 0 & 1 & 0 \\ \frac{\mu}{\lambda+\mu} & 0 & \frac{\lambda}{\lambda+\mu} \\ 0 & 0 & 1 \end{bmatrix} \qquad \Lambda = \begin{bmatrix} 2\lambda & 0 & 0 \\ 0 & \lambda+\mu & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\Rightarrow \Gamma = \begin{bmatrix} 2\lambda & 2\lambda & 0 \\ \mu & -\lambda-\mu & \lambda \\ 0 & 0 & 0 \end{bmatrix}$$

$$A = \begin{bmatrix} s+2\lambda & -2\lambda & 0 \\ -\mu & s+\lambda+\mu & -\lambda \\ 0 & 0 & s \end{bmatrix}$$

again, assume $\vec{P}(0) = [1\ 0\ 0]$

[redacted] PROCESSOR cont.

$$\det A = s(s+2\lambda)(s+\lambda+\mu) - 2\lambda\mu s$$
$$= s(s+r_1)(s+r_2)$$

$$r_1 = \frac{1}{2}\left(3\lambda+\mu+\sqrt{\lambda^2+6\lambda\mu+\mu^2}\right)$$
$$r_2 = \frac{1}{2}\left(3\lambda+\mu-\sqrt{\lambda^2+6\lambda\mu+\mu^2}\right)$$

$$A^{-1} = \begin{bmatrix} \frac{s(s+\lambda+\mu)}{s(s+r_1)(s+r_2)} & \frac{2\lambda s}{s(s+r_1)(s+r_2)} & \frac{2\lambda^2}{s(s+r_1)(s+r_2)} \\ \frac{\mu s}{s(s+r_1)(s+r_2)} & \frac{s(s+2\lambda)}{s(s+r_1)(s+r_2)} & \frac{\lambda(s+2\lambda)}{s(s+r_1)(s+r_2)} \\ 0 & 0 & \frac{s^2+(3\lambda+\mu)s+2\lambda^2}{s(s+r_1)(s+r_2)} \end{bmatrix}$$

$$\vec{P}(s) = P(0) \times A^{-1}$$

$$P_3(s) = \frac{2\lambda^2}{s(s+r_1)(s+r_2)}$$

DUAL PROCESSOR cont.

$$P_3(s) = \frac{A}{s} + \frac{B}{s+r_1} + \frac{C}{s+r_2}$$

by partial fraction expansion

$$A = 1$$

$$B = \frac{r_2}{r_1 - r_2} = \frac{3\lambda+\mu-\sqrt{\lambda^2+6\lambda\mu+\mu^2}}{2\sqrt{\lambda^2+6\lambda\mu+\mu^2}}$$

$$C = \frac{r_1}{r_2 - r_1} = \frac{3\lambda+\mu+\sqrt{\lambda^2+6\lambda\mu+\mu^2}}{2\sqrt{\lambda^2+6\lambda\mu+\mu^2}}$$

inverting the transform,

$$P_3(t) = 1 + Be^{-r_1 t} + Ce^{-r_2 t}$$

noting that the reliability $R(t) = 1 - P_3(t)$, and defining

$$Q = \frac{1}{2}(3\lambda+\mu)$$

$$R = \frac{1}{2}\sqrt{\lambda^2+6\lambda\mu+\mu}$$

$$R(t) = -\frac{Q-R}{2R}e^{-(Q+R)t} + \frac{Q+R}{2R}e^{-(Q-R)t}$$

DUAL PROCESSOR cont.

$$R(t) = \frac{Q-R}{2R} e^{-Qt} e^{-Rt} + \frac{Q+R}{2R} e^{-Qt} e^{Rt}$$

$$= e^{-Qt} \left[-\frac{Q-R}{2R} e^{-Rt} + \frac{Q+R}{2R} e^{Rt} \right]$$

$$= e^{-Qt} \left[\frac{Q}{2R} \left(e^{Rt} - e^{-Rt} \right) + \frac{1}{2} \left(e^{Rt} + e^{-Rt} \right) \right]$$

$$R(t) = e^{-Qt} \left[\frac{Q}{R} \sinh(Rt) + \cosh(Rt) \right]$$

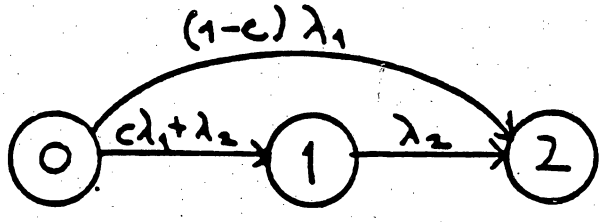
with $Q = \frac{1}{2}(3\lambda + \mu)$

$$R = \frac{1}{2}\sqrt{\lambda^2 + 6\lambda\mu + \mu^2}$$

COVERAGE

For real systems, fault diagnosis and isolation is far from perfect.

=> Coverage (usually denoted c) is defined as the conditional probability that a system recovers given there was a failure.



$$R_{sys} = R_1 + cR_2(1-R_1)$$

for $R_1 = R_2 = R$

$$R_{sys} = R + cR(1-R)$$

$$R_{sys} = (1+c)R - cR^2$$

for $c = 1$, $R_{sys} = 2R - R^2$

COVERAGE cont.

for a parallel system with n units (n-1 spares) each with identical reliability R and coverage c

$$R_{sys} = R \sum_{i=0}^{n-1} c^i (1-R)^i$$

evaluating the geometric progression,

$$R_{sys} = R \left(\frac{1 - c^n (1-R)^n}{1 - c(1-R)} \right)$$

$$R_{sys} = R \left(\frac{1 - (cQ)^n}{1 - cQ} \right)$$

EFFECT OF COVERAGE

for R = 0.9,

Table of R_{sys}

c \ n	2	4	∞
0.8	.9720	.9782	.9783
0.9	.9810	.9889	.9890
0.99	.9891	.9988	.9889
1.0	.9900	.9999	1.0000

$$MTTF(n \text{ modules}) = \int_0^\infty R(t) \sum_{i=0}^{n-1} c^i (1-R(t))^i \, dt$$

$$= MTTF(n-1 \text{ modules}) + \int_0^\infty R(t)\, c^{n-1} (1-R(t))^{n-1} \, dt$$

let $R(t) = e^{-\lambda t}$, and integrate,

$$= MTTF(n-1 \text{ modules}) + \frac{c^{n-1}}{n\lambda}$$

$$= \frac{1}{\lambda c} \sum_{i=1}^{n} \frac{c^i}{i}$$

[redacted]

$c = 1 \Rightarrow R_{sys} = 1 - Q^n = R_1(t)$

$c < 1 \Rightarrow R_{sys} = R\left(\frac{1 - cQ^n}{1-cQ}\right) = R_c(t)$ for some improvement factor I,

$$R_1(It) = R_c(t)$$

$$1 - Q(It)^n = R(t)\left(\frac{1-(cQ(t))^n}{1-cQ(t)}\right)$$

let $R(T) = e^{-\lambda T}, Q(T) = 1 - e^{-\lambda T}$

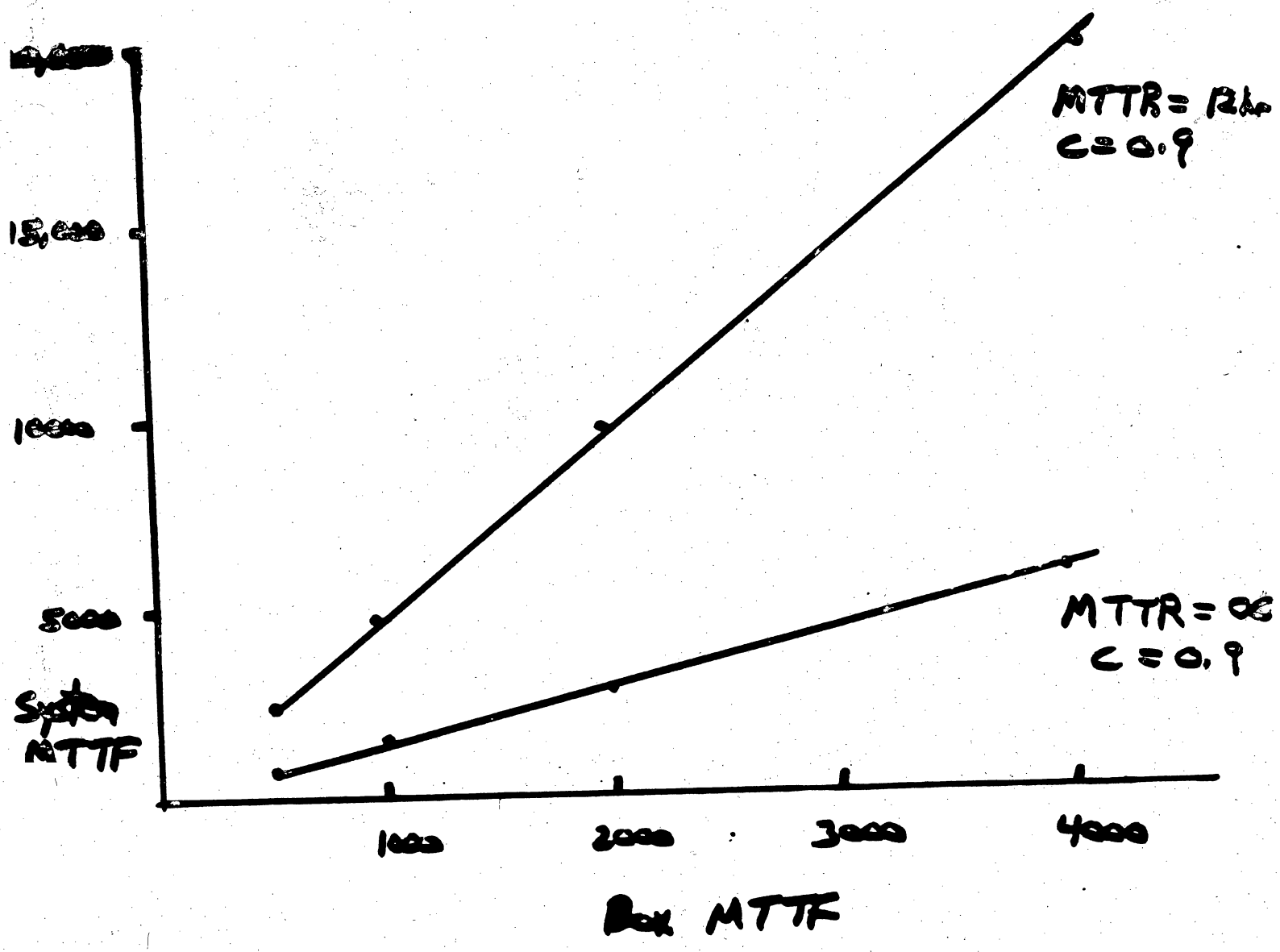
solve for I

$$I = -\frac{1}{\lambda t} \ln\left\{1 - \left[1 - R(t)\left(\frac{1-(cQ(t))^n}{1-cQ(t)}\right)\right]^{\frac{1}{n}}\right\}$$

for R = 0.9,

c \ n	2	4
0.8	1.738	4.601
0.9	1.408	3.720
0.99	1.047	1.957

Mission Time Improvement from increasing c to 1

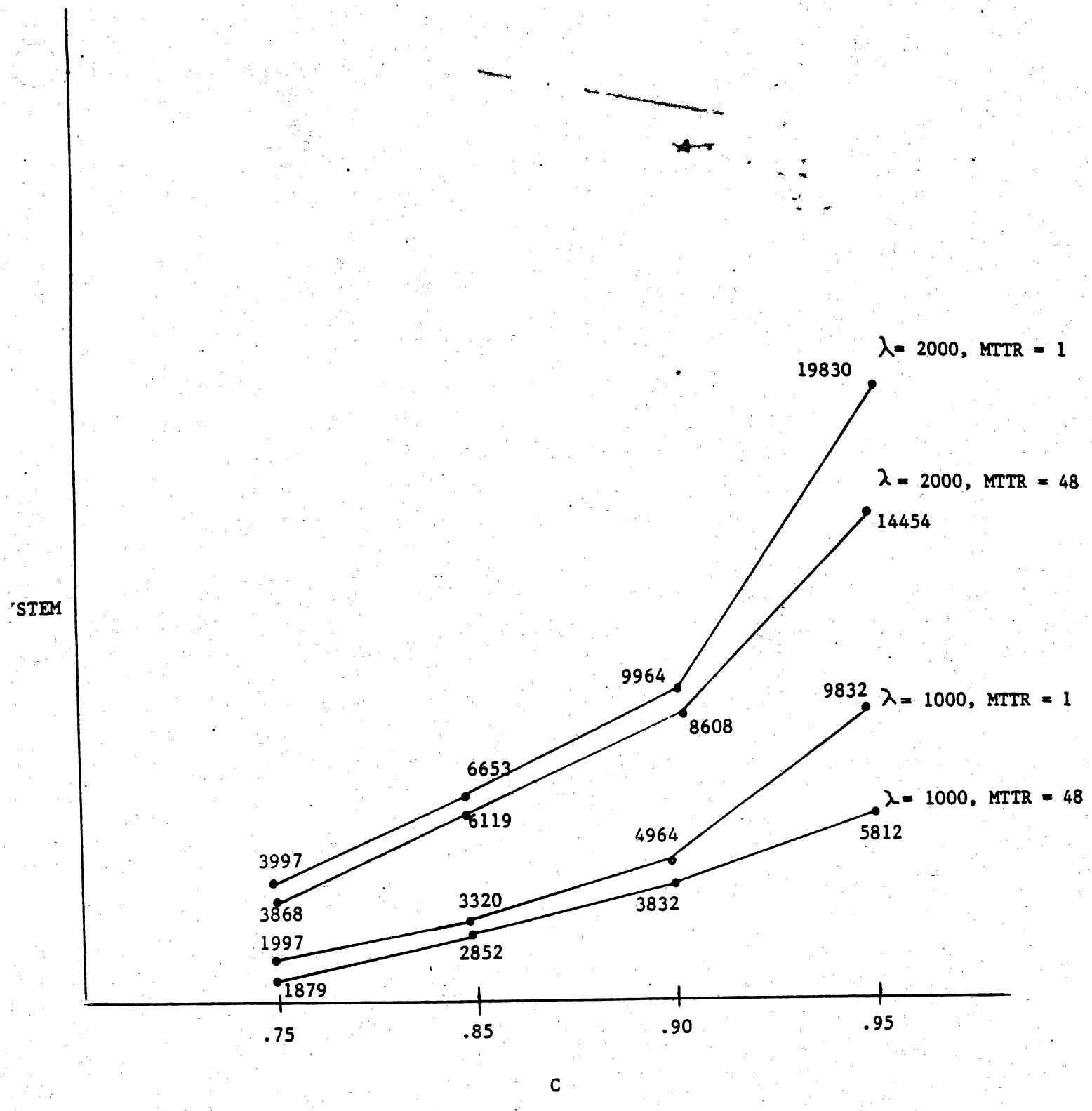


⇒ differences in models

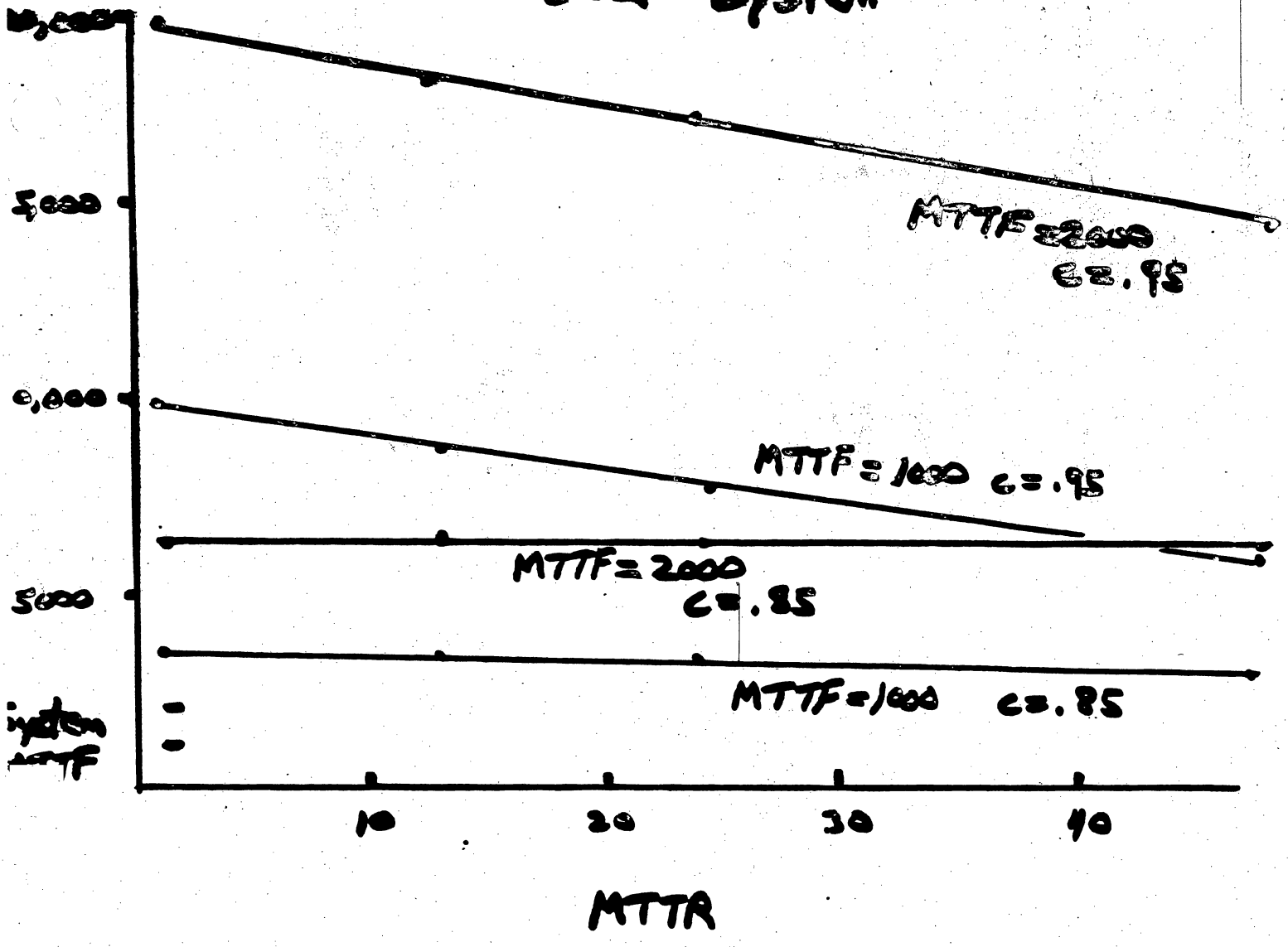
MTTR = ∞ easy

MTTR = finite exponentially hard

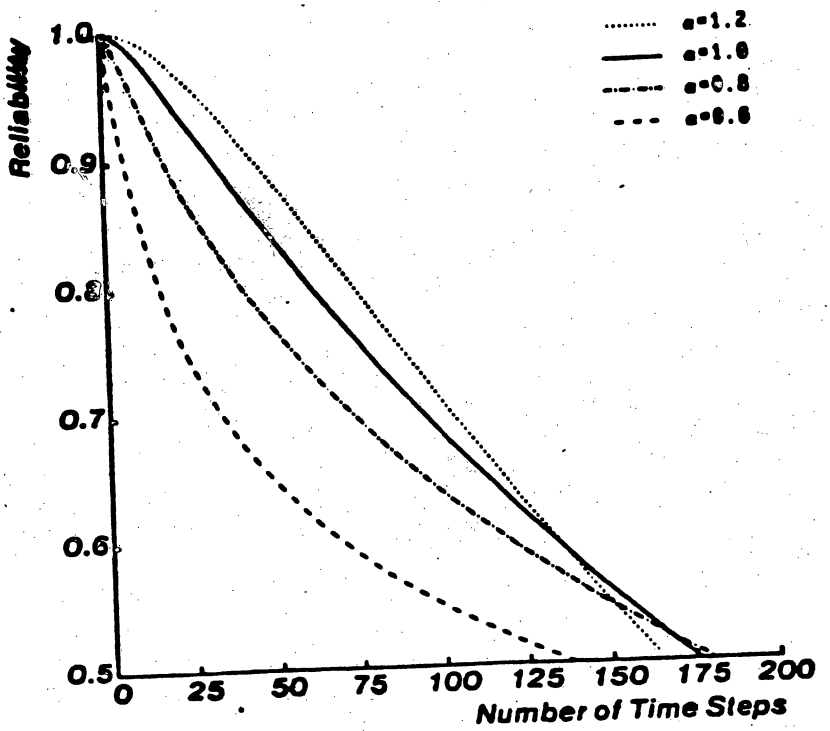
dual system a 1 week


λ= 2000, MTTR = 1
19830
λ= 2000, MTTR = 48
14454
'STEM
9964
9832
λ= 1000, MTTR = 1
8608
6653
λ= 1000, MTTR = 48
6119
4964
5812
3997
3868
3320
3832
1997
2852
1879
.75
.85
.90
.95
C

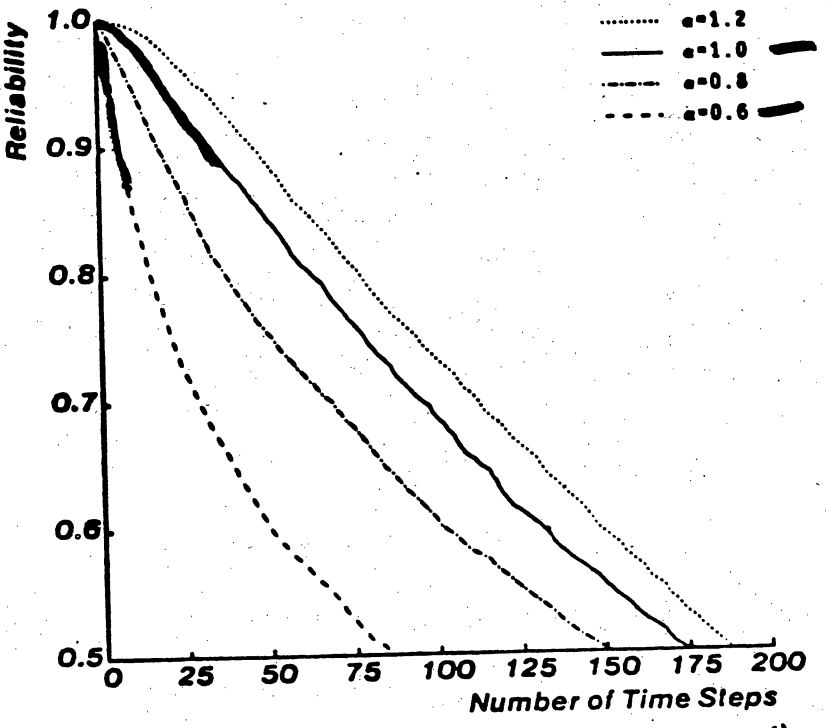
Dual System



⇒ c more important than MTTF



(a) Analytic Model Results (Without Renewal)



(b) Simulation Model Results (With Renewal)

Figure 5-3: TMR System Reliability

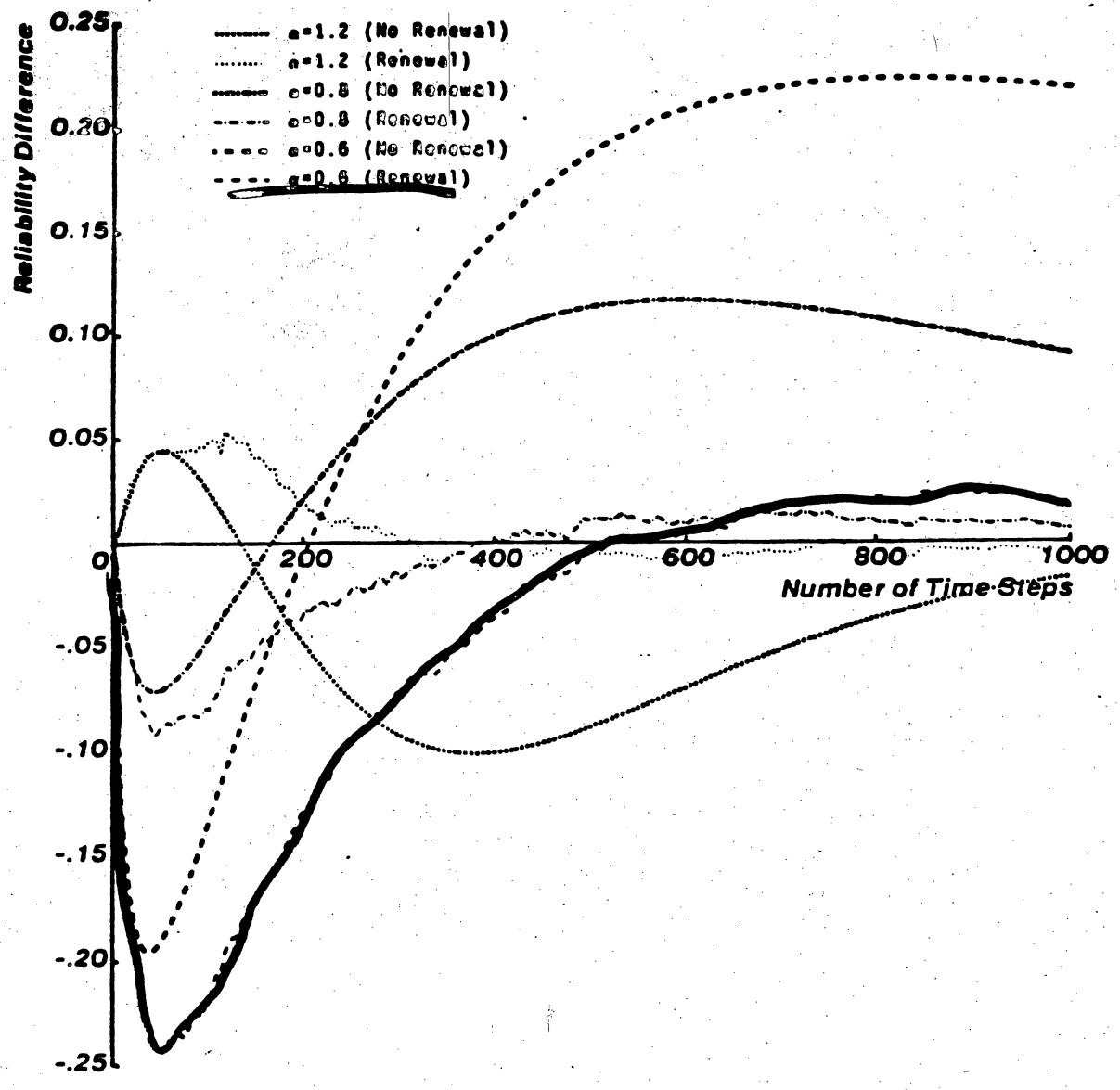


Figure 5-7: Reliability Difference of TMR System Relative to $\alpha_e = 1.0$

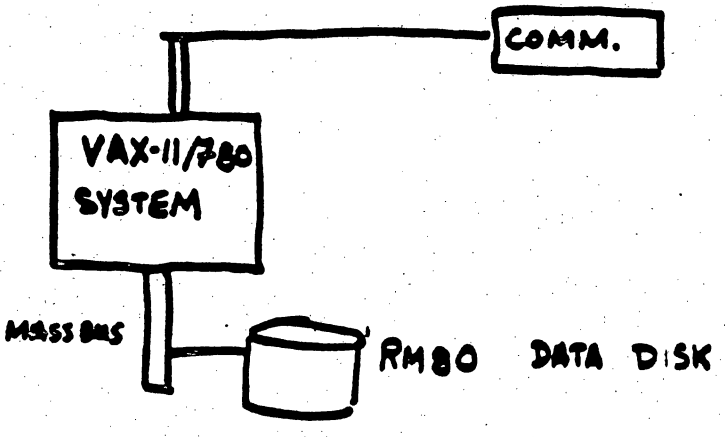
Example: Data collection system

System type: data collection system for oil drilling platform

MTTR: ~24 Hours

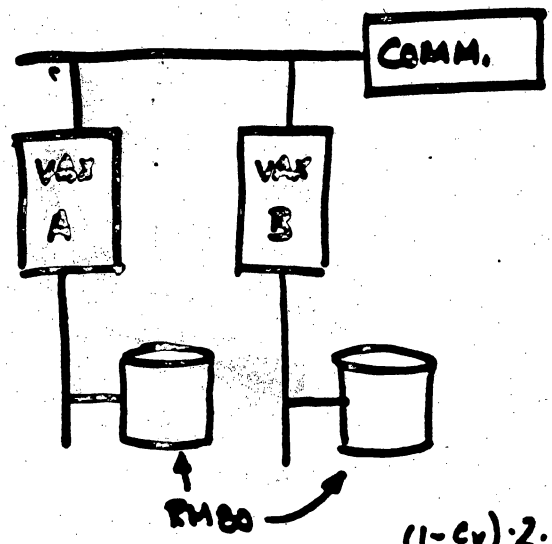
MTTF: 6 months

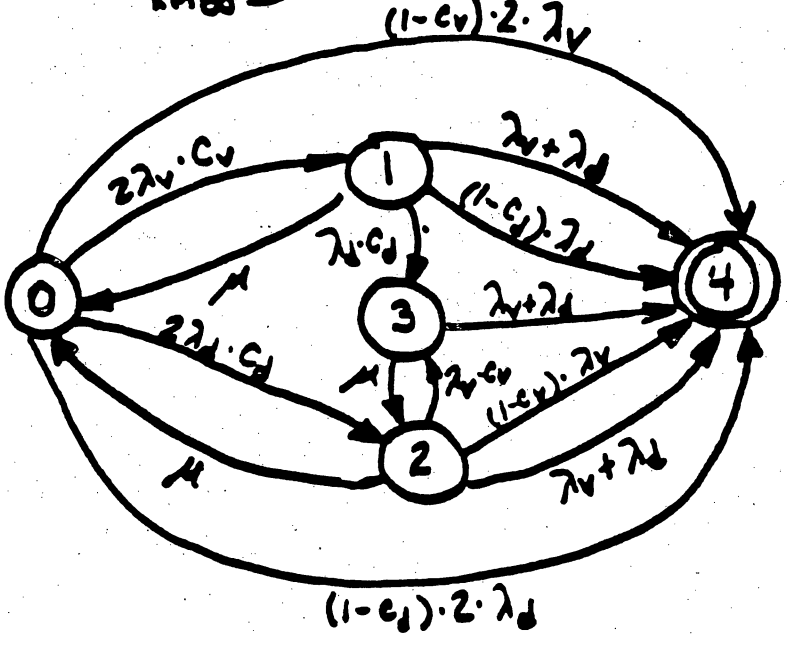
system configuration:



Results from Lambda:	1771 hours
(crosscheck with SEC4:	1802 hours)
Try *something:*	SEC control store
Lambda results:	1828 hours

Have to go to redundancy at system level





states:

0. all OK
1. one VAX down
2. one disk down
3. one disk, one VAX down
4. failed

Output from Steve's Markov Simulator
V1.3.0 (3-APR-85)
Address questions to: COIN::ELKIND

Program run on 21-APR-1985 at 04:25:12.56
Input file: THV3MKA.IN

---

first version - dual CPU, dual database disks

state assignment:
0 - all ok
1 - one Vax down
2 - one disk down
3 - one disk, other Vax
4 - failed

For transition matrix:

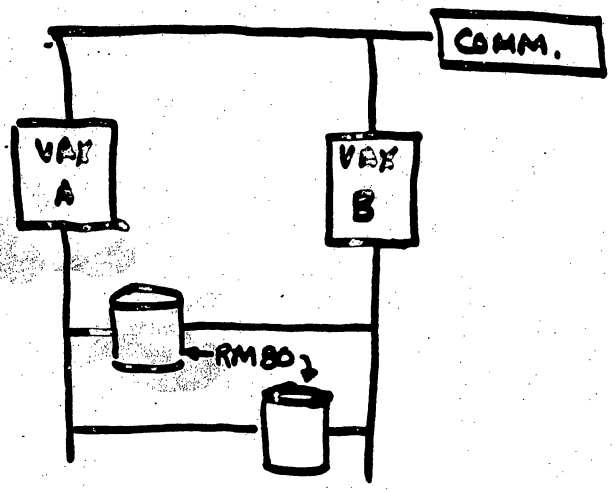
0.000E+00	9.735E-04	1.134E-04	0.000E+00	9.833E-06
4.167E-02	0.000E+00	0.000E+00	5.671E-05	5.484E-04
4.167E-02	0.000E+00	0.000E+00	4.867E-04	5.533E-04
0.000E+00	0.000E+00	4.167E-02	0.000E+00	5.484E-04
0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00

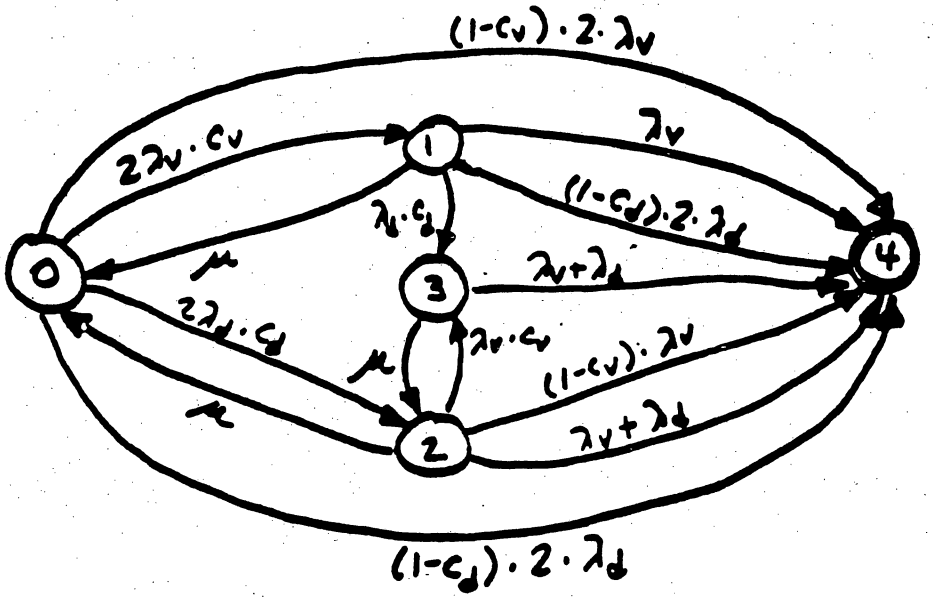
Number of simulations: 1000
Mean failure time : 43367.91
Standard deviation: 43153.7852

Transition count table over all simulations:

State	0	1	2	3	4	Total visits
0	0	40880	4805	0	399	46084
1	40304	0	0	44	532	40880
2	4780	0	0	59	68	4907
3	0	0	102	0	1	103
4	0	0	0	0	0	0

Simulation CPU time: 101.79 sec.





States:

0. all OK
1. one VAX down
2. one disk down
3. one VAX, one disk down
4. failed

first version - dual CPU, dual database disks

state assignment:

Mean failure time : 43367.91
Standard deviation: 43153.7852

Transition count table over all simulations:

State	0	1	2	3	4	Total visits
0	0	40880	4805	0	399	46084
1	40304	0	0	44	532	40880
2	4780	0	0	59	68	4907
3	0	0	102	0	1	103
4	0	0	0	0	0	0

Simulation CPU time: 101.79 sec.

---

second version - dual CPU, dual-ported database disks

state assignment:
0 - all ok
1 - one Vax down
2 - one disk down
3 - one disk, one Vax
4 - failed

For transition matrix:

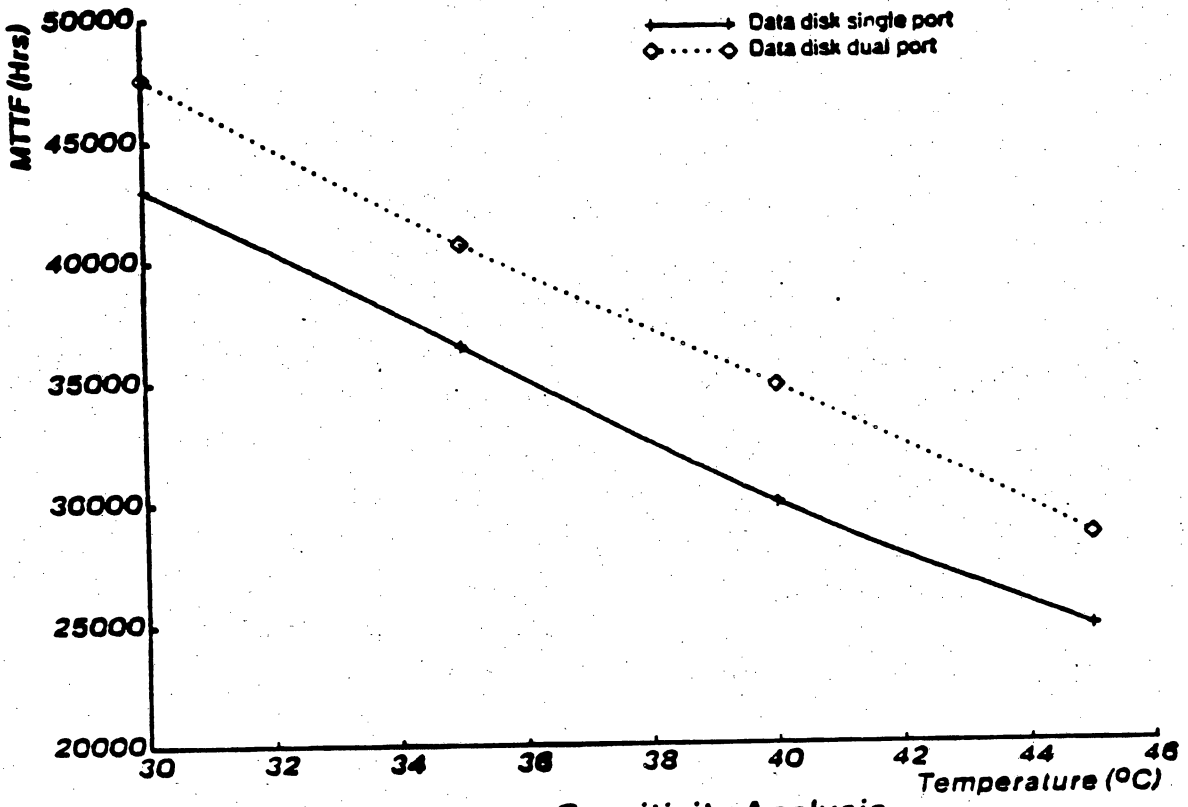
0.000E+00	9.735E-04	1.134E-04	0.000E+00	9.833E-06
4.167E-02	0.000E+00	0.000E+00	1.134E-04	4.917E-04
4.167E-02	0.000E+00	0.000E+00	9.735E-04	6.654E-05
0.000E+00	0.000E+00	4.167E-02	0.000E+00	5.484E-04
0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00

Number of simulations: 1000
Mean failure time : 48598.09
Standard deviation: 47001.8945

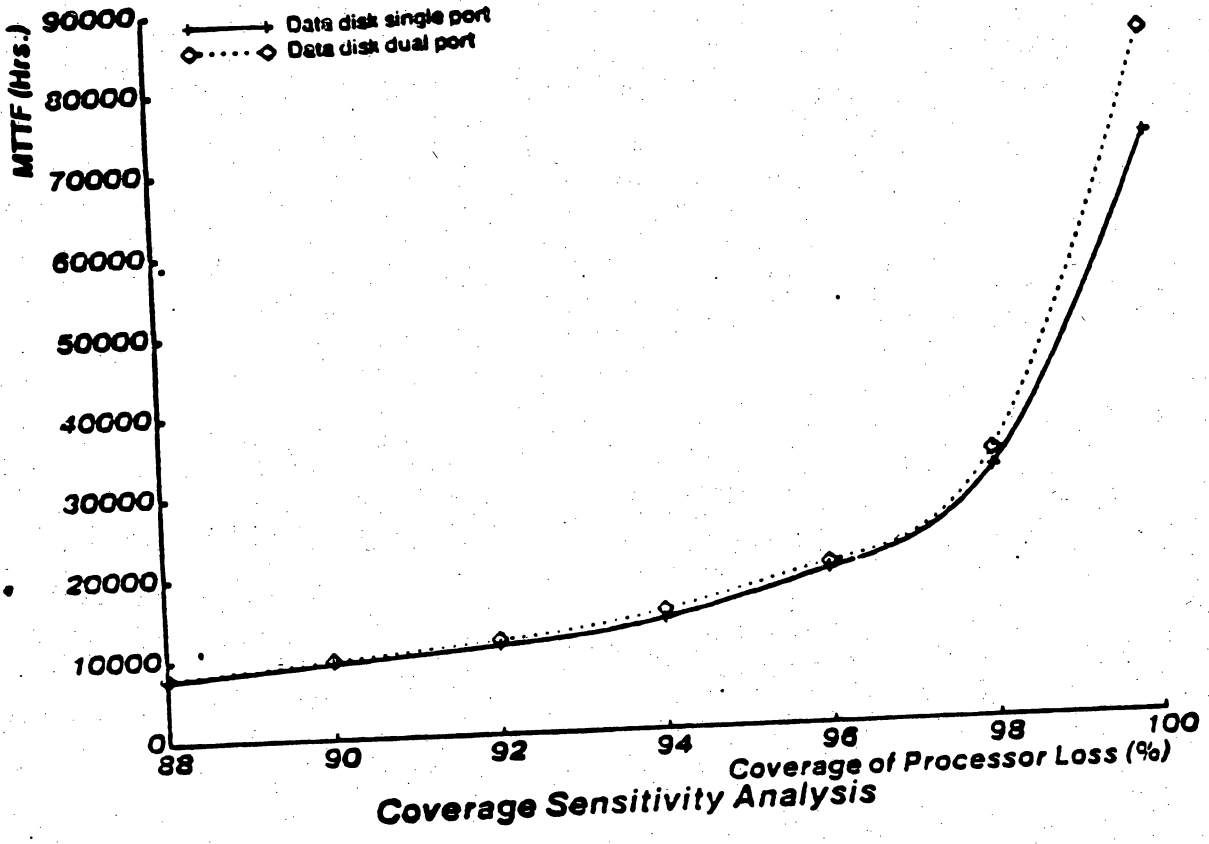
Transition count table over all simulations:

State	0	1	2	3	4	Total visits
0	0	45696	5348	0	457	51501
1	45034	0	0	132	530	45696
2	5467	0	0	131	7	5605
3	0	0	257	0	6	263
4	0	0	0	0	0	0

Simulation CPU time: 113.88 sec.



Temperature Sensitivity Analysis



Coverage Sensitivity Analysis

Reliability Evaluation: Available Modelling Tools

- Parts count models (ORACLE, ARPP)
- Combinatorial (CARE II, ADVISER)
- Markov models (ARIES 82)
- Semi-Markov models (CARE III, HARP)
- Event-driven simulator (SIMPLE)
- General modeller (RAT)

Modelling Tools Developed

Lambda

- Part failure rate calculator
  - MIL-217D
- Structured analyses of system reliability
  - top-down system description
  - failure rate analysis
  - MTTF analysis
  - hierarchical parameter selection
- Reliability "programming language"
  - general output formats
  - automated sensitivity studies
  - standard or user-supplied models of parts, part parameters, redundancy, etc.

Modelling Tools Developed (cont.)

MARK1

- Markov model simulator
- availability, MTTF simulation
- state transition count tables
- text file command driven

EC4

- hierarchical description of memory organization allowing mixed failure modes
- MTTF, R(t)
- text file command driven

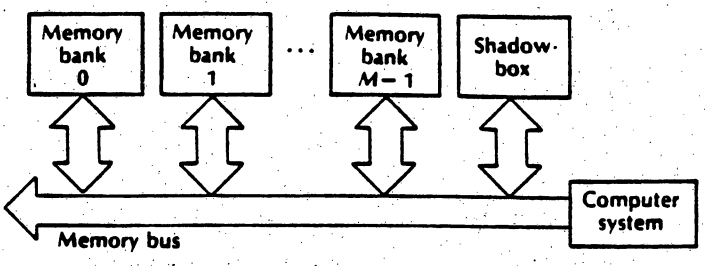


Figure 3–80. The shadow box memory backup technique proposed by Arulpragasm and Swarz.

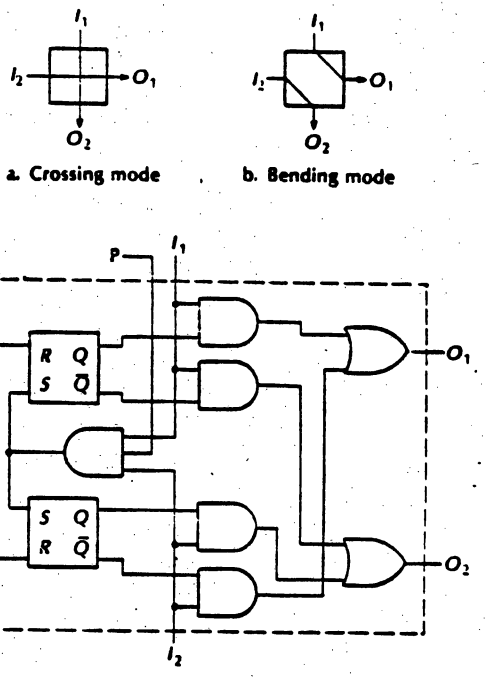


Figure 3–87. Basic two-input-to-two-output switching cell for implementing complex switches.

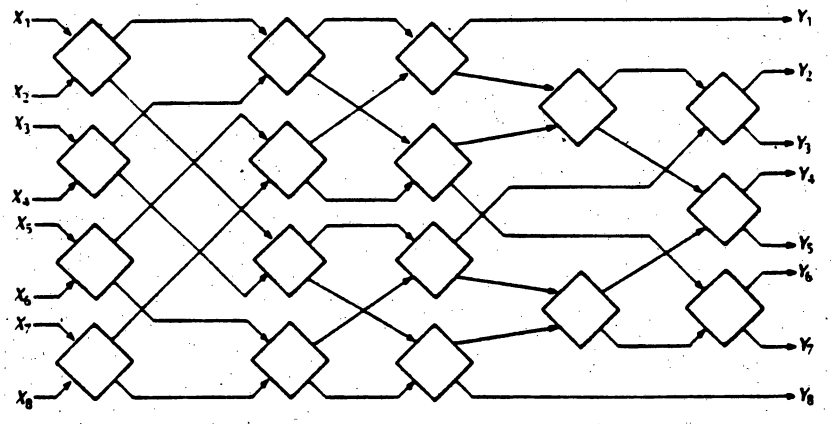


Figure 3–88. CPCU(8) switching network implemented from basic cells of Figure 3-87.

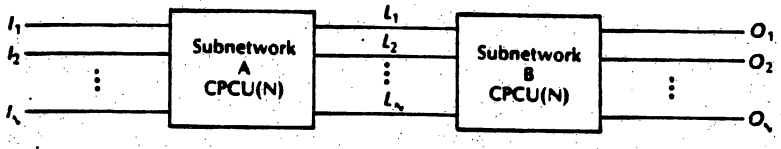


Figure 3–89. A fault-tolerant network, in which the damage due to a single stuck-at fault in one subnetwork can be compensated for by the other subnetwork.

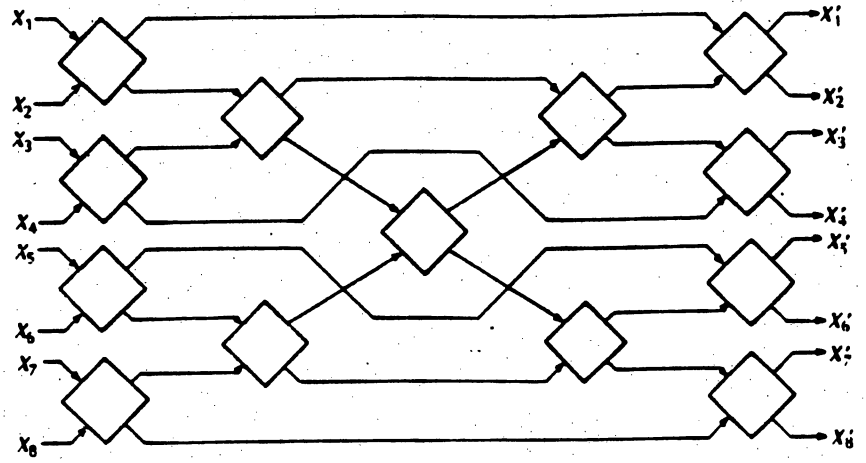


Figure 3–90. Compensation network used as subnetwork A (Figure 3-89) for a CPCU(8) single stuck function fault-tolerant network.

Recovery
Retry

- transient error solution

- Ethernet
- busses

UNIVAC 1100/60
programmable pause

- instruction retry
- process retry

⇒ rollback

Alternate Data retry

Catalog of Software Reliability Techniques

Region	Techniques
Fault Avoidance	Modularity
	Object Oriented
	Capabilities
	Formal Proofs
Masking Redundancy	Diverse Programming
Dynamic Redundancy	Forward Error Recovery
	Backward Error Recovery
	- Journaling
	- Checkpointing
	- Recovery Blocks

Fault Avoidance

Modularity — Information Hiding

Defensive Programming

Object Oriented — Protection on object

Export Functions

Objective C, C++ Front ends to C

Capability Based — Protection on access path

Advantage - Reusable Software

Disadvantage - Overhead

- "Architectural Discipline"

Fault Avoidance

Formal Proofs

High level Fault models

- Fail stop
- Fail benign
  - lost messages
  - delayed messages
- Byzantine failure

Disadvantage -

By construction - overhead
- example SIFT

By proof - beyond state of art
- example SIFT 900 lines of Pascal

SOFTWARE

MENU OF RELIABILITY TECHNIQUES

- Modular Programming - General
- Preventive Maintenance - General
- Error Reporting - General
- Regression Testing - Common
- Design Reviews - Great variation
- Interactive debug - universal
- Desk Checking - Common
- Peer code review - occasional
- Change review - rare
- Acceptance Checking - Rare
- Top-down design - unknown
- Program Design Language - rare
- Test coverage analyzer - rare
- Assertion checker - unknown
- HIPO - unknown
- Flowcharts - unknown
- Test driver - common
- System modeling, simulation - rare

Management and Organization

Quality assurance organization - not in traditional sense.
Product test organization - rare.
Configuration management organization - rare.
Chief programmer team - rare.
Test plans/procedures/reports - Sometimes.
Reliability modeling to estimate test completion - rare.

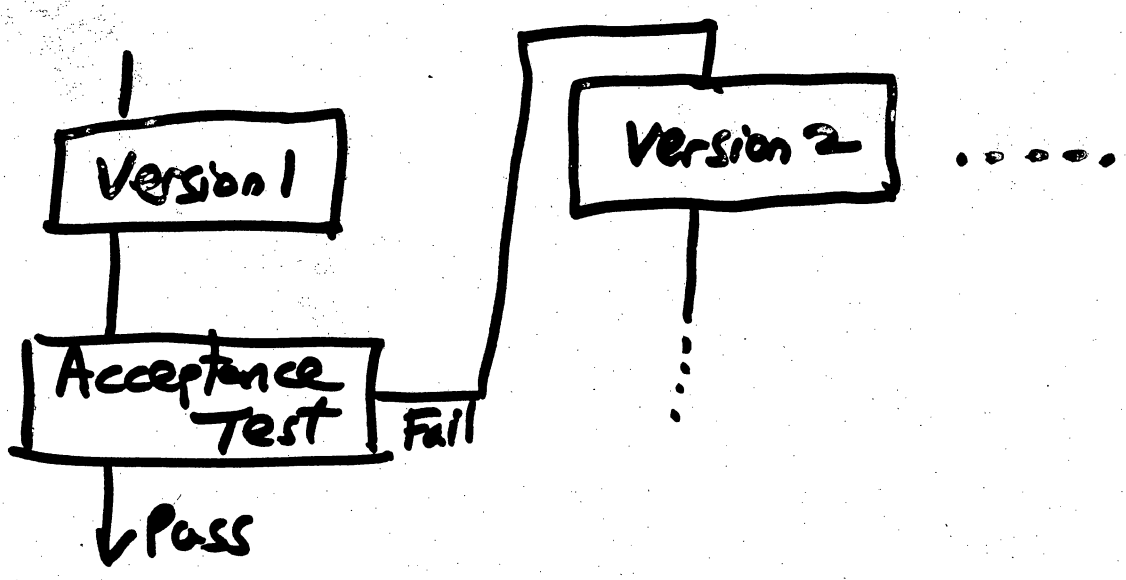
Masking Redundancy

Diverse Programming — Multiple programming teams

Example - Space Shuttle

Disadvantage - expensive, multiple programming teams
- specification still a single source of failure

Recovery Block



- Versions can be separately developed or earlier release of software

Advantage - with hardware assist for block structured language can make recovery automatic
- example, LSI-11 recovery cache, compare to C.mmp

Disadvantage - develop fast, effective acceptance tests

Experiences with Recovery Blocks

Phase	
1	1st operational release
2	Upgraded recovery mechanisms
3	2nd release
4	Rewritten modules
5	Pre-release version

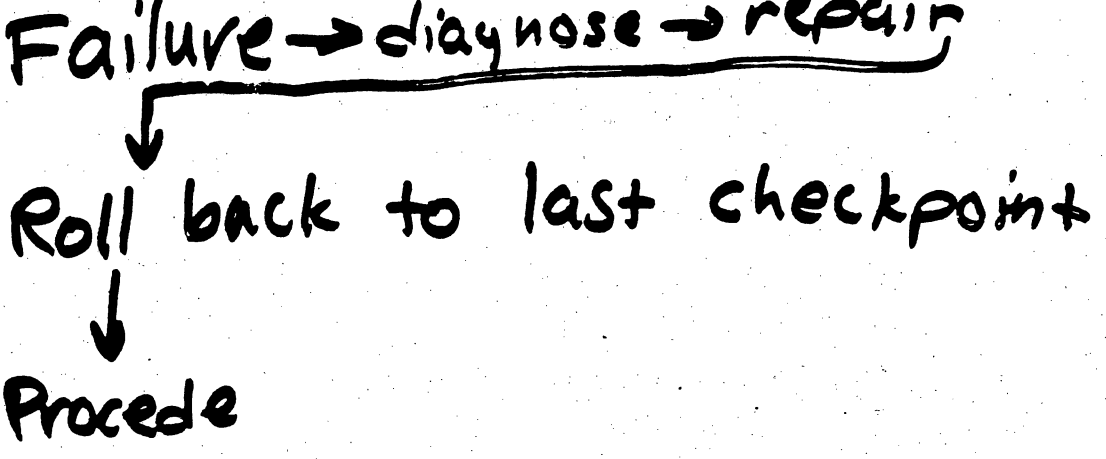
Phase	Coverage	90% Confidence	Aborted Coverage
1	.75	.67	.68
2	.60	.52	.53
4	.81	.77	.81
	.74	.72	.71

165/222 Recovery mechanisms

157/222 caused

Checkpoint & Rollback

Periodically, store all state information needed to successfully complete task.



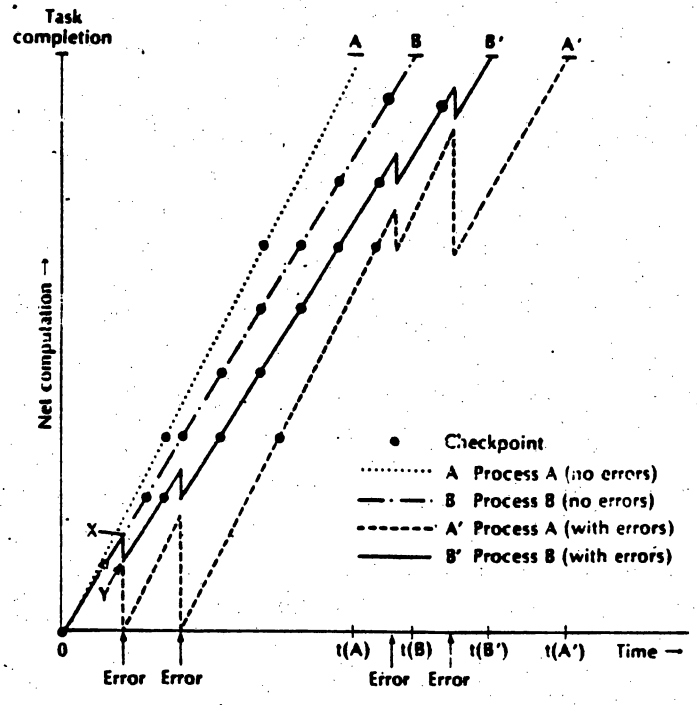


Figure 3–91. Scenario of two processes, identical except for checkpoint frequency.

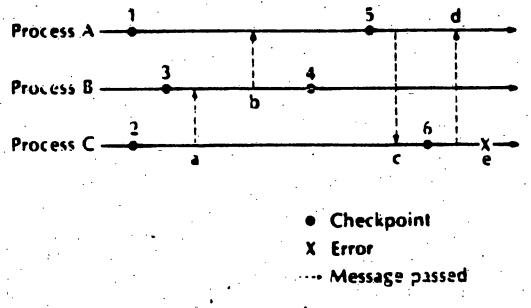
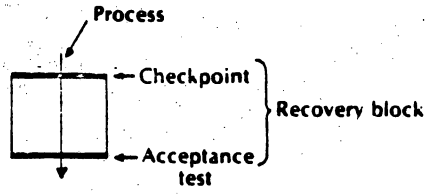
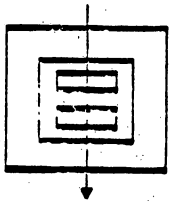


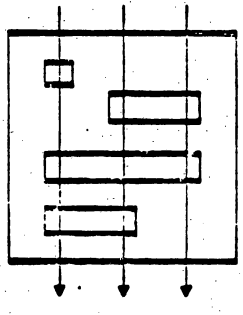
Figure 3–92. Cooperating checkpointing processes. A failure at point *e* forces process C to roll back to checkpoint 6. Because of a message sent by process C, process A must then be rolled back to checkpoint 5. Rolling back of the processes in this fashion eventually requires all three processes to be rolled back to their initial checkpoints.



a. Graphic notation for a recovery block



b. Nesting of recovery blocks



c. Multiprocess recovery blocks

Figure 3–94. Use of Randell's graphical notation to demonstrate extensions beyond the simple recovery block.

Journalling

Save initial state,
record list of
commands/changes/operations

Failure → diagnose → repair

↓

start from beginning,
recreate state just
before failure

e.g., Bravo

ce

Graceful Degradation

Allow system capacity to Degrade in the event of failures, but still operate.

- can use "excess" to boost capacity before failures occur.
- useful computing performed a factor

Examples:

SIFT, FTMP, PLURIBUS
Cm*, C.mmp, TANDEM — Software use of "HOOKS"
VAX-11/780, 1100/60

Support for Reliability Modelling Concepts

Concept	*Packages Supporting*
Measures	
λ	Lambda
MTTF	Lambda, SEC3, SEC4, MARK1
MT(r)	Lambda
MTI(r)	(Lambda)
R(t) (and MR(t))	Lambda, SEC3, SEC4
A	MARK1 (Lambda)
MTTC	MARK1
MTBPR	Lambda
MTBF, MTBC	none
Repair	
Modelling	none
Effect on Reliability	MARK1 (Lambda)
ault Detection Coverage	
Modelling	none
Effect on Reliability	MARK1 (Lambda)
Fault Handling Coverage	
Modelling	(MARK1)
Effect on Reliability	MARK1 (Lambda)
Transient/Intermittent Failures	
Modelling	none
Effect on Reliability	MARK1
General z(t)	
Modelling	none
Effect on Reliability	(Lambda)
Diagnosability	none
Performability	none

Lambda Library Functions and Procedures

Name	*Type*	*Purpose*
Reliability Models		
REL_TMR	function	TMR module reliability (majority-vote)
REL_NMR	function	NMR module reliability (majority-vote)
M_OF_N	function	*m*-out-of-*n* modules reliability
SEC_MEM	function	SEC code memory reliability
ECC_MEM	function	General ECC code memory reliability
DUAL	function	Reconfig. duplication, with coverage
DUAL_REPAIR	function	Reconfig. duplication, with repair
Miscellaneous		
IC_THETA	function	Θ_{ja} model for DIPs
CUSTOMLAMBDA	function	Mil-217D model for PC board λ
MTTF_STATS	procedure	Print statistics on MTTF integration

Modelling Support of Reliability Enhancement Techniques

Technique	*with no repair*	*with repair*	*coverage modelling*
Fault Avoidance (all)	Lambda*	n/a	n/a
Fault Detection (all)	none	n/a	n/a
Masking Redundancy			
NMR/voting	Lambda*, MARK1 (SEC3)	MARK1 (Lambda)	n/a
ECC Codes			
Hamming code	SEC3, SEC4, Lambda*	none	n/a
Other codes	Lambda*	none	n/a
Masking Logic	(Lambda)	none	n/a
Dynamic Redundancy			
Reconfig. Duplication	MARK1, Lambda*	MARK1, Lambda*	(MARK1)
Reconfig. NMR	MARK1 (Lambda)	MARK1 (Lambda)	(MARK1)
Backup Sparing	MARK1, Lambda*	MARK1 (Lambda)	(MARK1)
Graceful Degradation	MARK1 (Lambda*)	MARK1 (Lambda)	(MARK1)

Modelling Tool Experience

Package	*Projects*
FAIL	VAX-11/750, VAX-11/730
Lambda	VAX-11/785, VAX-8600, SAI C^3, PDP-11/23+ CPU Board System projects A, B, C, D, E, G Design systems D0 (D1 & MICON)
SEC3	Trilogy logic modules
SEC4	System project D
MARK1	System projects E, F

5.1) Overview of Testing Theory

Table 4–1. Stages in the development of a system.

Stage	*Error Sources*	*Error Detection Techniques*
Specification and design	Algorithm design	Simulation
	Formal specification	Consistency checks
Prototype	Algorithm design	Stimulus/ response testing
	Wiring and assembly	
	Timing	
	Component failure	
Manufacture	Wiring and assembly	System testing
	Component failure	Diagnostics
Installation	Assembly	System testing
	Component failure	Diagnostics
Operational field	Component failure	Diagnostics
	Operator errors	
	Environmental factors	

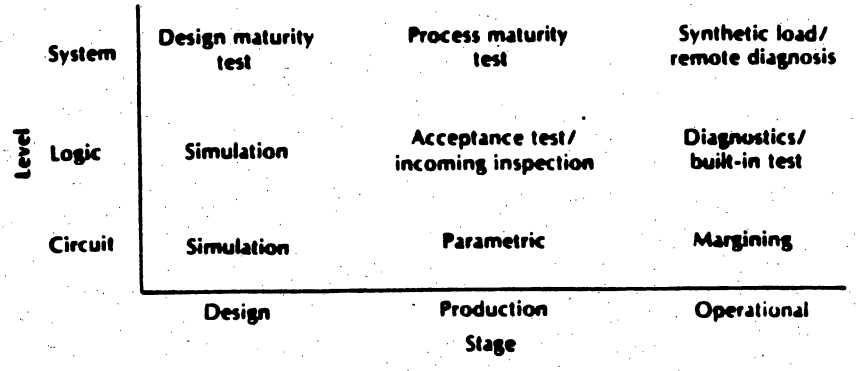


Figure 4–1. Testing as a function of system level and time.

Table 4–2. Typical MOS parametric tests.

Gate-oxide breakdown voltage
Drain-to-substrate breakdown voltage
Drain-to-source punchthrough voltage
Gate-to-source threshold voltage
Drain current at 0 gate voltage
Drain current at specified operating voltage
Gate-to-source leakage current
Drain-to-substrate leakage current
Transconductance at specified operating voltage
Drain-source resistance

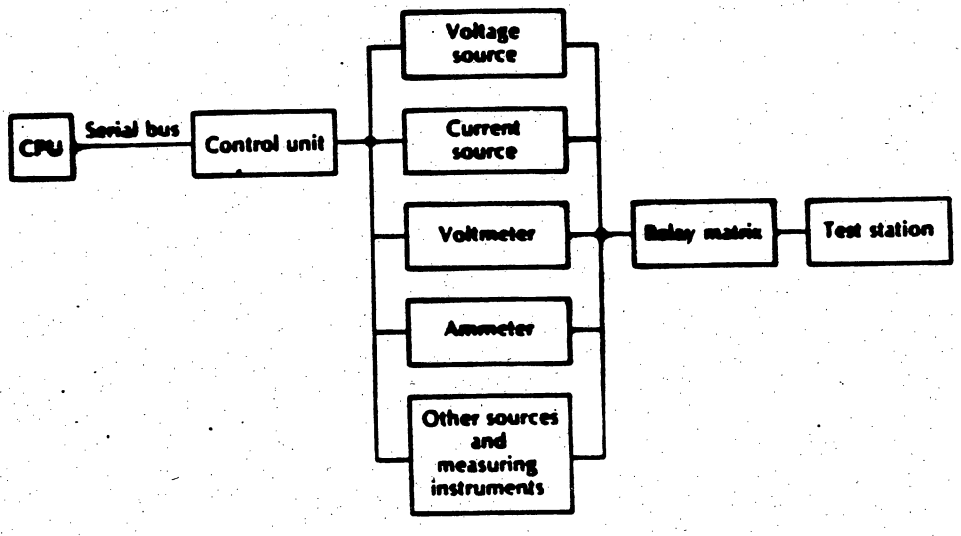


Figure 4–3. Block diagram of an automated parametric test system.

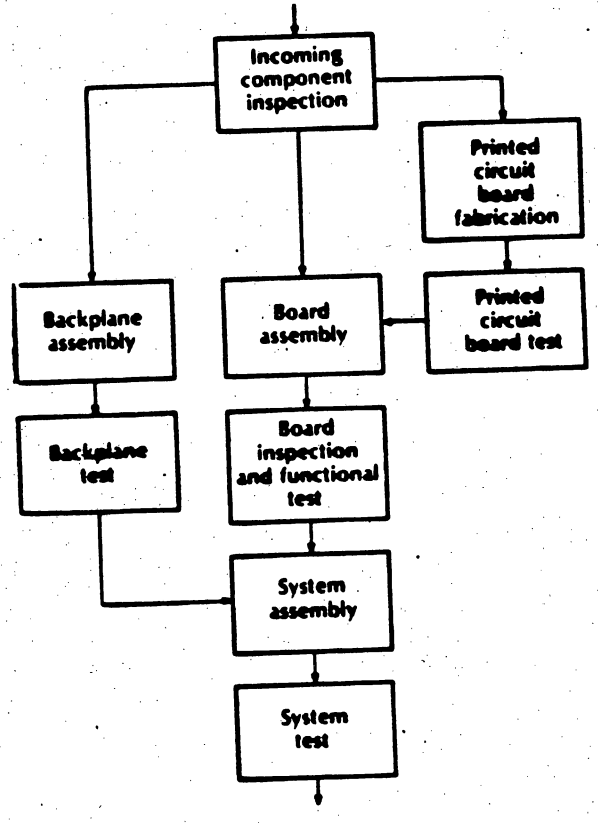


Figure 4–2. Typical steps in the manufacture of a digital system. (© 1979 IEEE.)

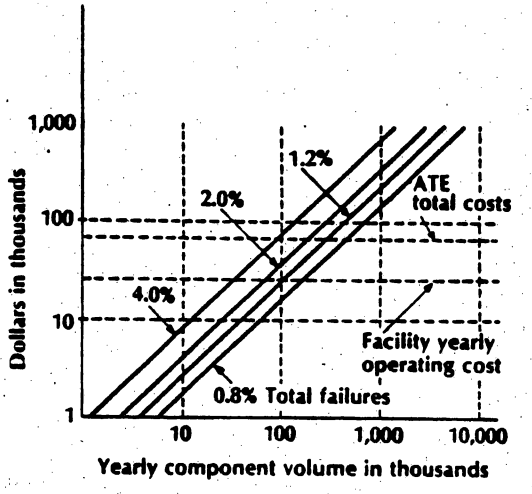


Figure 1–6. Savings from screening and testing as a function of defective component rate and annual device volume.

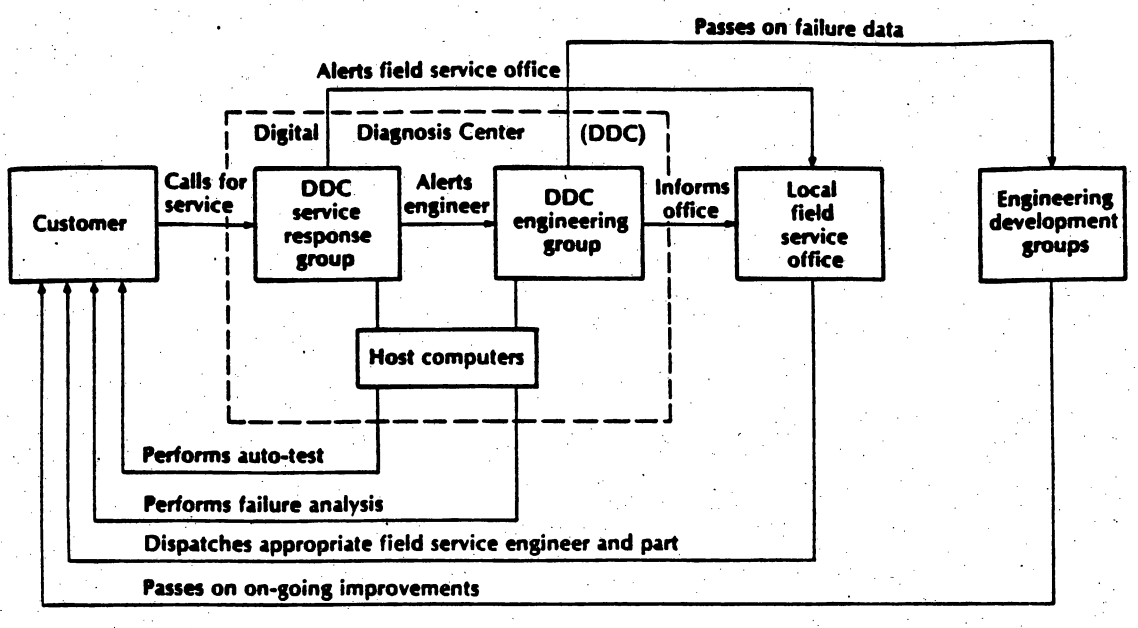
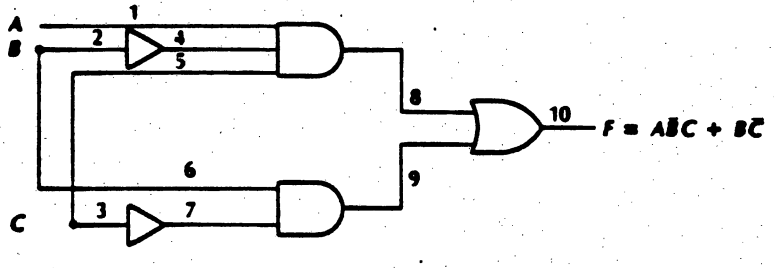


Figure 1–7. Overview of DEC's Remote Diagnosis Network.

Table 4–3. Approaches to stimulus generation and response checking.

Stimulus Generation	*Response Checking*
Exhaustive	Exclusive OR
Random	Stored
	Compact testing
	Transition counting
	Signature analysis
Stored	
Simulation	Predicted response
Deductive	
Parallel	Fault dictionary
Concurrent	
Algorithmically generated	
Algebraic	
Boolean difference	
Path sensitization	
D-algorithm	
On-chip	On-chip



Test set

A	B	C	F
1	0	0	0
0	0	1	0
1	0	1	1
0	1	0	1
1	1	1	0

Figure 4–4. A circuit for test generation.

Table 4–4. The *D*-algorithm definition of elementary gate functions in terms of the symbol *D*.

AND		
Input 1	*Input 2*	*Output*
1	1	1
1	0	0
0	1	0
0	0	0
1	D	D
D	1	D
1	$\overline{D}$	$\overline{D}$
$\overline{D}$	1	$\overline{D}$
0	D	0
D	0	0
0	$\overline{D}$	0
$\overline{D}$	0	0
D	D	D
$\overline{D}$	$\overline{D}$	0
D	$\overline{D}$	0
$\overline{D}$	D	0

OR		
Input 1	*Input 2*	*Output*
1	1	1
1	0	1
0	1	1
0	0	0
1	D	1
D	1	1
1	$\overline{D}$	1
$\overline{D}$	1	D
0	D	D
D	0	D
0	$\overline{D}$	$\overline{D}$
$\overline{D}$	0	$\overline{D}$
D	D	D
$\overline{D}$	$\overline{D}$	$\overline{D}$
D	$\overline{D}$	1
$\overline{D}$	D	1

Inverter	
Input	*Output*
1	0
0	1
D	$\overline{D}$
$\overline{D}$	D

	Line									
Step	1	2	3	4	5	6	7	8	9	10
Initial test on Line 5	x	x	x	x	$\bar{D}$	x	x	x	x	x
Implication on other gate inputs	x	x	$\bar{D}$	x	$\bar{D}$	x	x	x	x	x
Propagate to Line 8	x	x	$\bar{D}$	x	$\bar{D}$	x	x	$\bar{D}$	x	x
Implication on other gate inputs	1	0	$\bar{D}$	1	$\bar{D}$	x	x	$\bar{D}$	x	x
Propagate to Line 10	1	0	$\bar{D}$	1	$\bar{D}$	x	x	$\bar{D}$	x	$\bar{D}$
Implication on other gate inputs	1	0	$\bar{D}$	1	$\bar{D}$	0	0	$\bar{D}$	0	$\bar{D}$

a. Forward propagation and implication

A	B	C	F
1	0	$\bar{D}$	$\bar{D}$

b. Test

Figure 4–5. The *D*-algorithm applied to Line 5 stuck-at-1 in Figure 4-4.

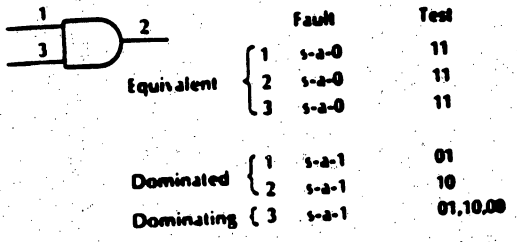


Figure 4–6. Equivalence and dominance relations among faults.

Table 4–5. Tests for pattern sensitivities in memory chips. (The test complexity is given in terms of the number of memory bits.)

Test	*Complexity*
Checkerboard pattern of 1s and 0s	N
Walking pattern	$N^{3/2}$
Galloping 1s and 0s (dynamic test)	N^2
Ping pong	N^2

Larry Lai

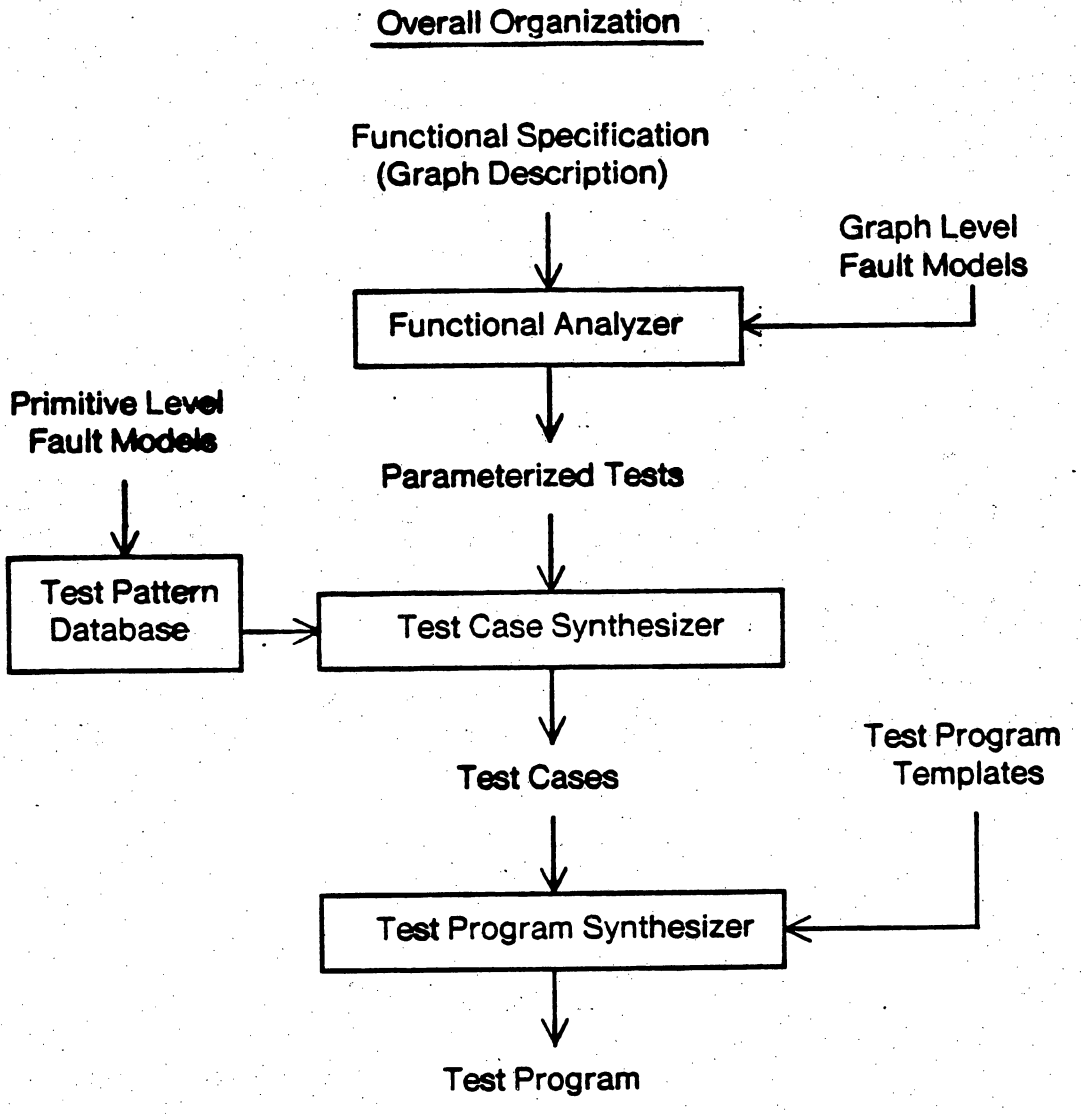


Figure 1-1: Overall Organization of Methodology

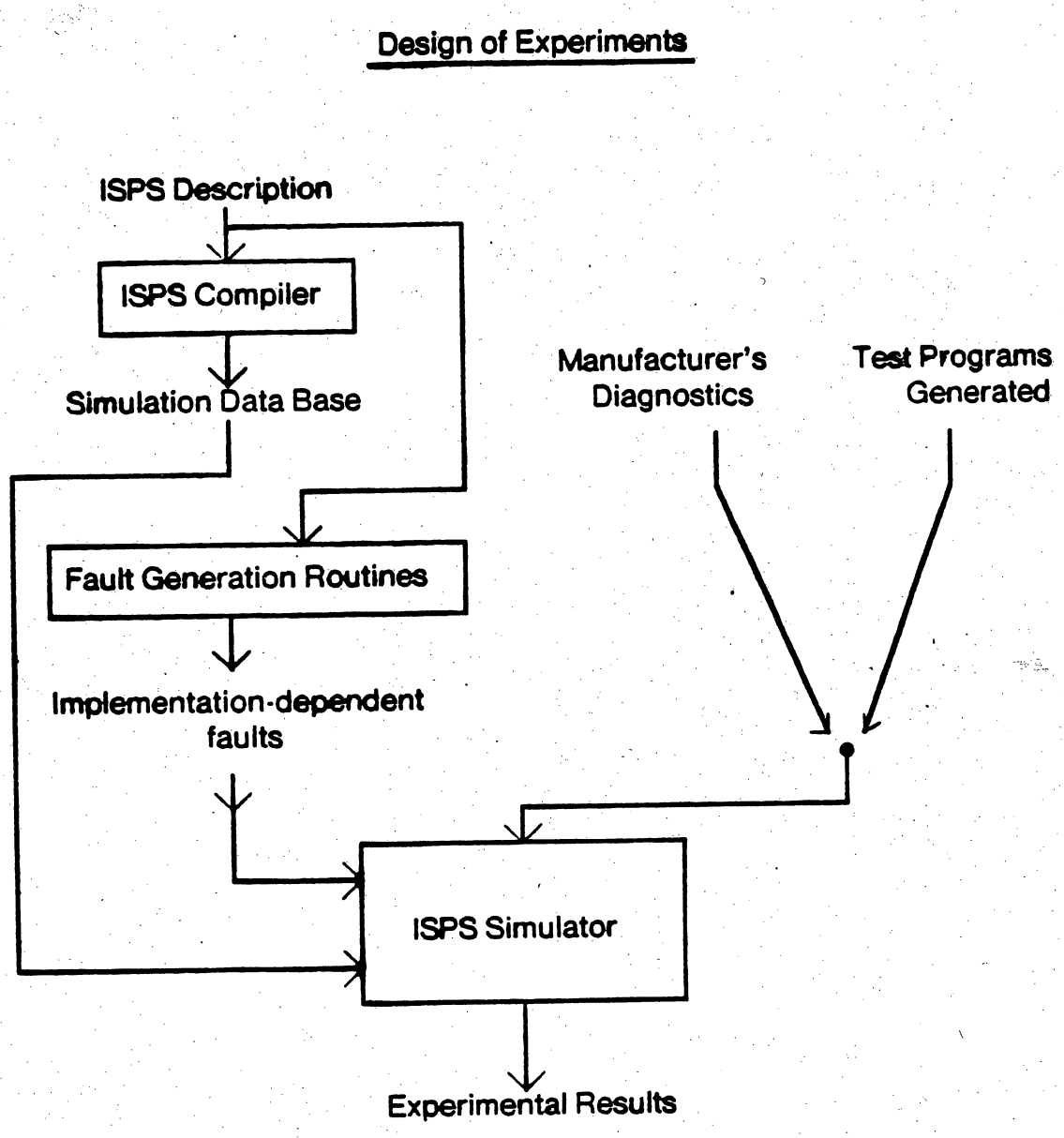


Figure 6-1: Design of Experiments

Diagnostics	Inst1	Inst2	Inst2B(i)	Inst2B(ii)	Ranjms	Isz	Ranjmp
Faults Simulated	1438	547	162	88	71	71	67
Faults Detected	891	385	74	17	0	4	2
Remaining Undetected Faults	547	162	88	71	71	67	65
Cumulative % of Faults Detected	62.0%	88.7%	93.9%	95.1%	95.1%	95.3%	95.5%
Execution Count	1835	1152	1536*	1536*	1536*	1536*	5120*
Cumulative Execution Count	1835	2987	4523	6059	7595	9131	14251
Lines of Code	2217	1711	682	(i + ii)	276	292	199

Test Number	1	2	3	4	5	6	7	8	9	10
Faults Simulated	1438	691	354	236	78	64	55	42	35	29
Faults Detected	747	337	118	158	14	9	13	7	6	7
Remaining Undetected Faults	691	354	236	78	64	55	42	35	29	22
Cumulative % of Faults Detected	52.0	75.4	83.6	94.6	95.6	96.2	97.1	97.6	98.0	98.5
Execution Count	60	68	113	62	50	45	122	83	95	33
Cumulative Execution Count	60	128	241	303	353	398	520	603	698	731
Paths Tested	5	6	11	4	3	3	11	6	3	1
Cumulative Paths Tested	5	11	22	26	29	32	43	49	52	53

Figure 6-2: Results of the Evaluation Experiment

Extrapolation Assuming N^2 Growth in Runtime

Speed up factor	1 hour	1 day	1 week	Ratio to PDP-8 (1 week)
1	393	1,927	5,098	1
5	1,967	9,635	25,491	109
10	3,933	19,269	50,981	217

Figure 6-4: Projected number of paths that can be handled by the system

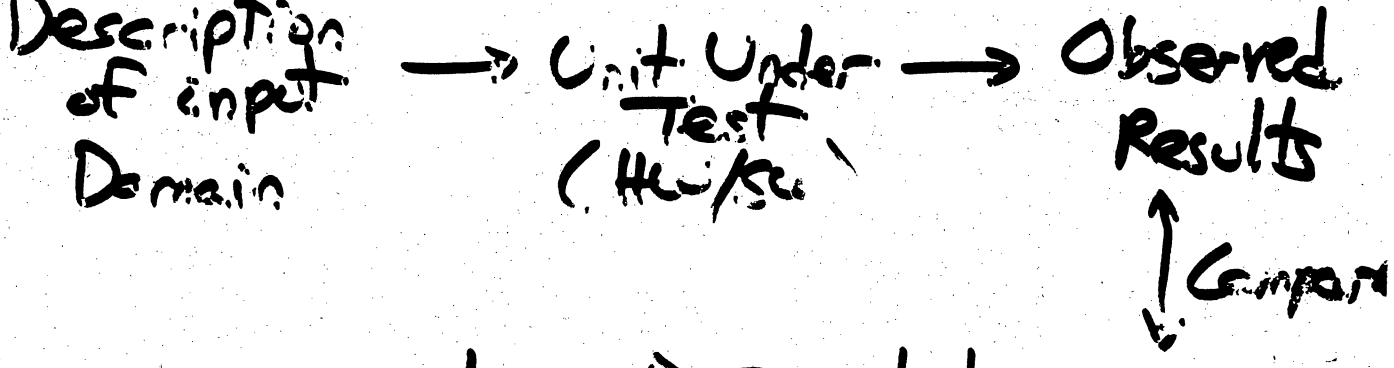
experiment-time.draw

	Digital System Tested			
	Full PDP-8	Reduced PDP-8	MARK1	JK flip flop
# of Nodes	158	76	50	16
# of Paths	235	120	77	18
Ratio(# of paths)	13.06	6.67	4.28	1
# of Data Path Tests Generated				
successful	66	37	21	3
failed	2	2	2	0
total	68	39	23	3
Heuristics Success Rate %	97	95	91	100
CPU Time(min.) with Interpreted LISP Code	21:25	6:51	1:19	0:08
Ratio-total CPU Time	160.6	51.4	9.88	1
Time per Path in system (sec)	5.47	3.43	1.03	0.44
Ratio - Time per Path	12.43	7.80	2.34	1
Time per Test Generated (sec)	18.90	10.54	3.43	2.67

Figure 6-3: Timings of the Automatic Test Generation System

Validation, Verification, and Testing of Software

Good For Software or Hardware during requirements and design stages.



Static and Dy[illegible]

Techniques

Manual

- Bench checking
- Peer Review
- Walkthroughs, inspection

Proof of Correctness

- validate assertions
- Formal Proof
  - logic representat
    - Separate fron code
    - Integrated with code
- Heuristics
  - array bounds checks
  - conflicts over common data

Techniques (cont.)

Simulation

Functional (Black Box)

- Boundary Value Analysis
  - zero, positive/negati
    crossings/maximum
- Design-based
  - using design
    hierarchy
- Cause-Effect Graphing
  - test cases From
    high level specifica
    eg. 8 stimuli ⇒
    256 "types" of inpu
    Classify response
    into effects class
    Decision table —
    causes 2, 3 result
    in effect 4

Techniques (cont.)

Structural (White Box)

- Discover test data with good coverage
  - Coverage-based
    - percent covered
  - Complexity-based
    - proportional to software complexity

- Coverage
  - branches
  - paths
  - decision-to-decision path
  - Finite domain testing number of iterations

Techniques (cont.)

- Complexity based
  - based on software science or graph complexity

Analysis of Test Data

- Estimate number of errors by number uncovered
- Seed known errors
- Mutation analysis
  - modify single statements
  - assumes program nearly correct
  - assumes single error test sets can uncover multiple errors

Static Analysis

Flow Analysis

- undefined variables

Symbolic execution

- with loops grows exponentially
- construct verification conditions

Dynamic Analysis

Counters to test branches taken

5.2) Design for Testability and Maintainability

Table 4–6. Stages in design for testability.

Stage	*Combinational*	*Sequential*
Test set for unmodified circuit	Structural faults	Extension of combinational approaches for structural faults
		Functional faults
Minimum modification to existing circuit	Add a small number of test points	Add synchronizing sequence
		Add distinguishing sequence
		Break selected feedback
Extensive modification to existing circuit	Improve controllability	Make combinational LSSD
	Improve observability	
New design	Reed-Muller expansion	Fail-safe design
	Totally self-checking circuits	

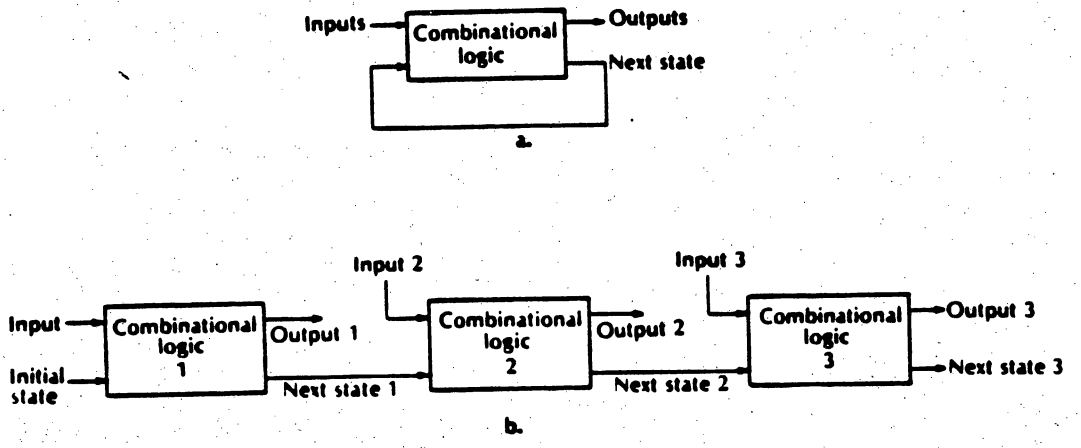


Figure 4–7. A sequential circuit replicated three times as a combinational circuit.

Table 4–7. Design for testability-added hardware types.

Test points
Edge connectors
Dual In-line Package (DIP) sockets
Terminal posts
Tristate drivers
IC clips
Pull-up resistors
Pin amplification
Input demultiplexers
Output multiplexers
Parity trees
Blocking gates
Control and observation switching
Disconnection structures
Edge connectors
DIP sockets
Tristate drivers
Blocking gates
Test-state register
Power-up reset
Scan-in/scan-out shift registers

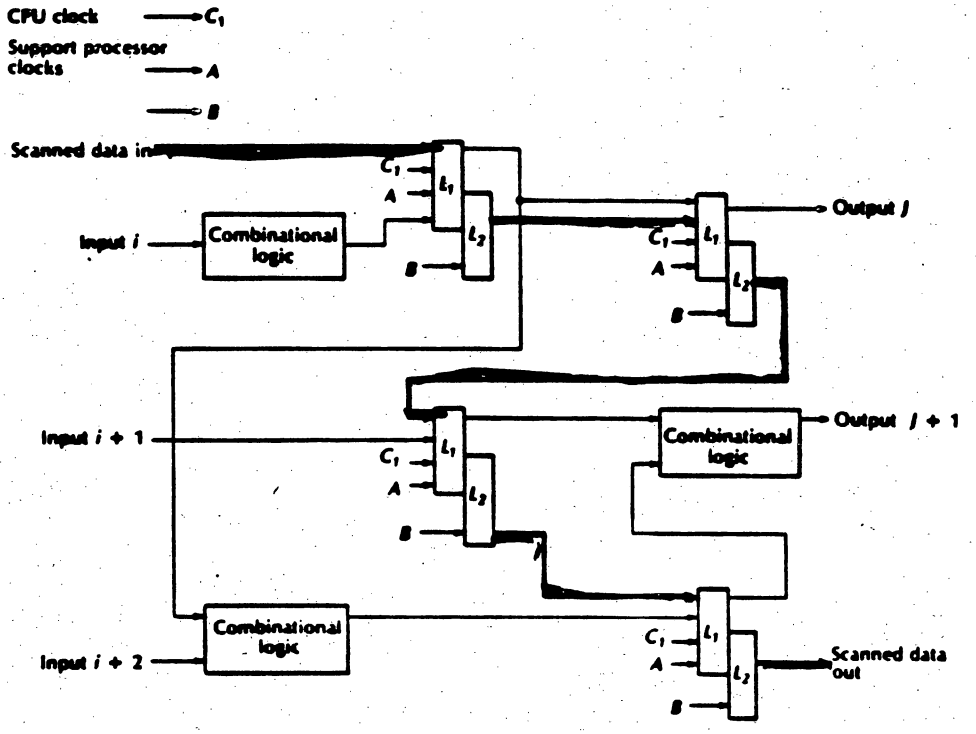


Figure 4–8. An example of LSSD.

Table 4–8. Design for testability-added hardware location suggestions.

1. Make sequential circuit components such as counters, shift registers, and control flip-flops initializable. Some ways of providing initializability are to wire control signals or testpoints to component clear or preset inputs, or to provide direct-load capabilities. Do not tie both the set and preset inputs of flip-flops to a common permanent logic signal.
2. Make counter chains controllable and observable in a reasonably short test sequence. For example, break long counter chains during test mode by inserting testpoints In the carry-propagate/count-control lines. This is especially important in the case of clock countdown circuits that are used to provide control inputs for the rest of the circuit. In the latter case it may even be wise to provide testpoints to bypass the counters entirely during portions of the test.
3. On-board clock oscillators should be made disconnectable during test. This can be done by disconnecting their output with a testpoint or by socketing them for removal during test.
4. If one-shots are used, control and observe their outputs with testpoints.
5. Try to break global feedback loops during test mode. Blocking gates can be used for this, rather than more costly testpoints.
6. Use added hardware to partition the circuit into functionally independent subcircuits for testing. This is especially important for separating digital and analog subcircuits. One method is to place testpoints between subcircuits.
7. Break reconvergent fan-out paths when they interfere with testability.
8. Place testpoints at locations of high fan-out or high fan-in.
9. Route logic drives of lamps and displays to testpoints so that the tester can check for correct operation. Make keyboard and switch outputs accessible to the test machine by breaking with testpoints.
10. In circuits containing microprocessors and other LSI devices, use testpoints to enhance controllability and observability of address buses and data buses, important control signals such as the reset and hold inputs to the microprocessor, and bus tristate control. In particular, the address and data terminals of RAMs and ROMs should be easily accessible.

Table 4–9. Design for testability suggestions not requiring added hardware.

1. Avoid the use of asynchronous sequential circuits. Edge-triggered *D*-type flip-flops are preferable to other types of flip-flops. These are synchronous, and behave merely as clocked data delays during testing.
2. Avoid one-shots when possible.
3. Avoid unnecessary wired-OR or wired-AND connections. When these must be used, try to employ gates from the same IC package to enhance fault locations.
4. Use elements in the same IC package when designing a series of inverters.
5. Try to assign gates in a feedback loop to the same IC package.

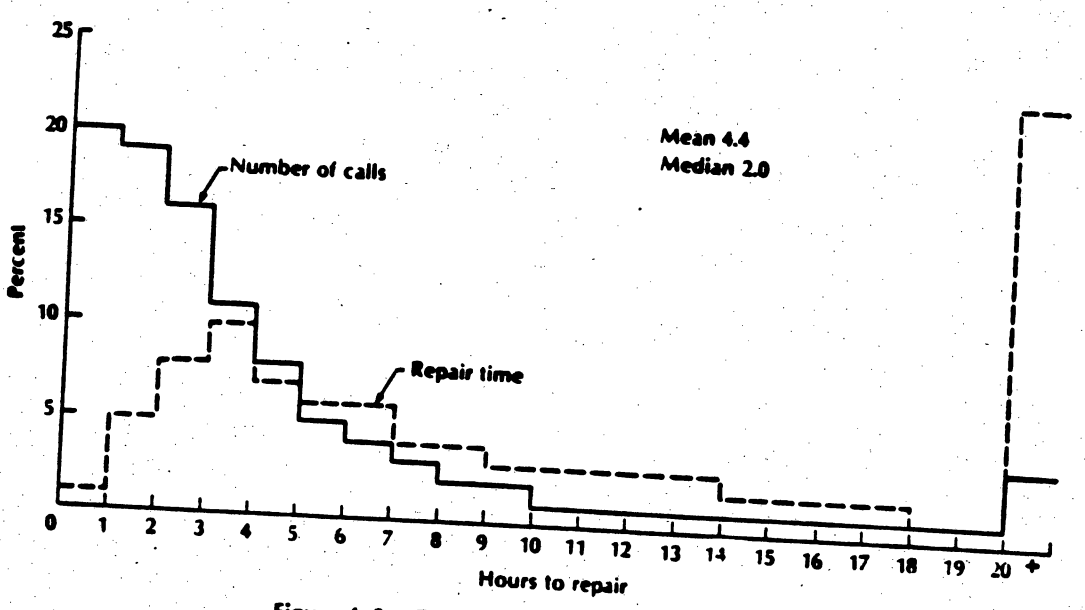


Figure 4–9. Time To Repair (TTR) distribution.

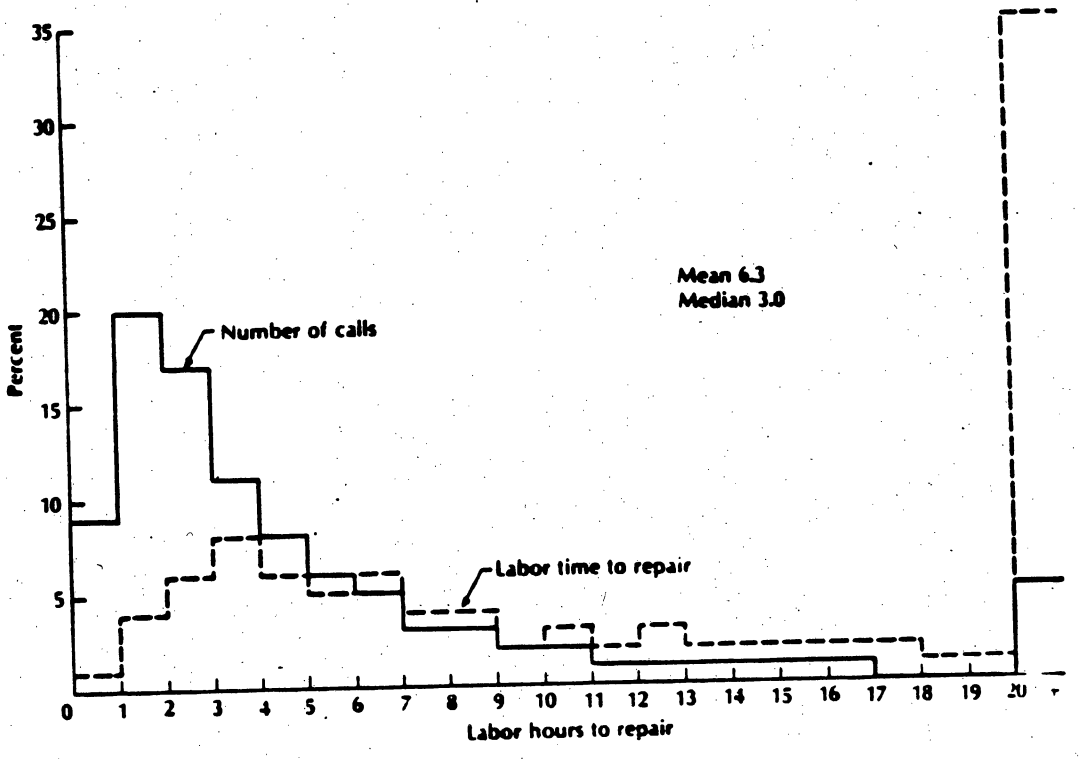


Figure 4–10. Labor Hour to Repair (LH) distribution.

Table 4–10. Suggestions on design for maintainability.

1. Start small. Verify subsystem operation step by step, from the minimal logic configuration through the addition of each incremental function.
2. Provide isolated environments so that each subsystem is completely exercised without requiring other subsystems.
3. Provide subsystem self tests, including tests of error-detection circuitry.
4. Provide internal observability and controllability in subsystems.
5. Provide error reporting and logging.
6. Minimize the need for external test equipment [illegible]
7. Provide a cabinet structure that facilitates repair.
8. Base repair strategies on component replacement, not on component swap.
9. Provide a support processor and remote access for diagnosis.

FAILURE RESPONSE STAGES

Fault Confinement

Fault Detection

Fault Masking

Retry

* Diagnosis

Reconfiguration

Recovery

Restart

Repair

Reintegration

DIAGNOSIS

To identify a condition by its signs, symptoms, or distinguishing characteristics.

To infer a process that generated a set of symptoms, outcomes, or results.

Diagnosis is *not* testing. Diagnosis is a problem-solving procedure whose goal is to guide the administration of tests.

Testing is a procedure whose goal is to provide information sufficient for determining whether or not a unit under test responds to a stimulus in accordance with a prespecified standard.

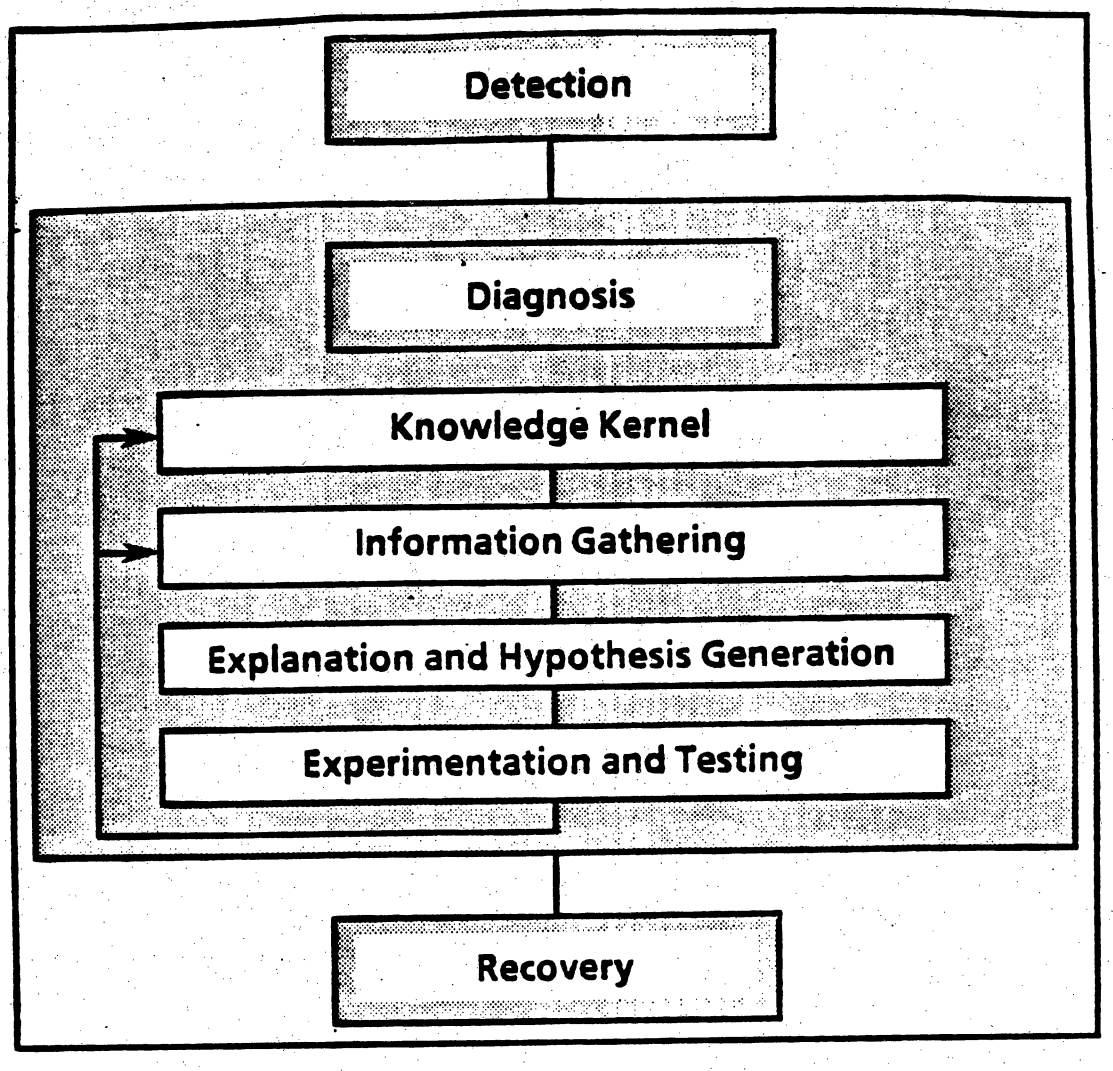
ASPECTS OF DIAGNOSIS

Retrospective diagnosis

Predictive diagnosis

Diagnosis as search

Diagnosis is not testing



The Diagnostic Cycle

Issues in Diagnostic Systems

Representation

Knowledge acquisition

Causal models

Device structure and function

Symptom-directed vs specification-directed

Human factors & cognitive aspects

Generalization & scaling up

Autonomy vs heteronomy

Dealing with uncertainty

Testing and validation

Tools and environments

```
V A X / V M S   SYSTEM ERROR REPORT   COMPILED 17-JUN-1985 14:39
********************* ENTRY     16. ***************************
ERROR SEQUENCE 199.                          LOGGED ON SID 02005EFF
MACHINE CHECK, 31-MAR-1985 22:59:02.04 KA750 REV# 255. MIC# 94.
     EXCEPTION PC    80008E57
     ERROR PSL       04070004
                                  Z-BIT
                                  INTERRUPT PRIORITY LEVEL = 07
                                  PREVIOUS MODE = KERN
                                  CURRENT MODE  = KERNEL
                                  INTERRUPT STACK
     SUMMARY CODE    00000002
                                  TRANSLATION BUFFER OR BUS ERROR
     VA LAST REF     80060243
     PC AT ERROR     80008E57
     MDR             0000005D
     SMR             00000008
                                  CPU MODE = KERNEL
                                  VIRTUAL
                                  READ
     RLTO            00000000
     TBER            00000003
                                  TB G0 DATA ERROR
                                  TB G1 DATA ERROR
     CAER            00000001
                                  CACHE HIT
     BER             00000000
     MCESR           00000006
                                  XB REFERENCE
                                  TB PARITY ERROR
```

Event Log

ANALYZING EVENT LOGS

Information Reduction

Information Organization

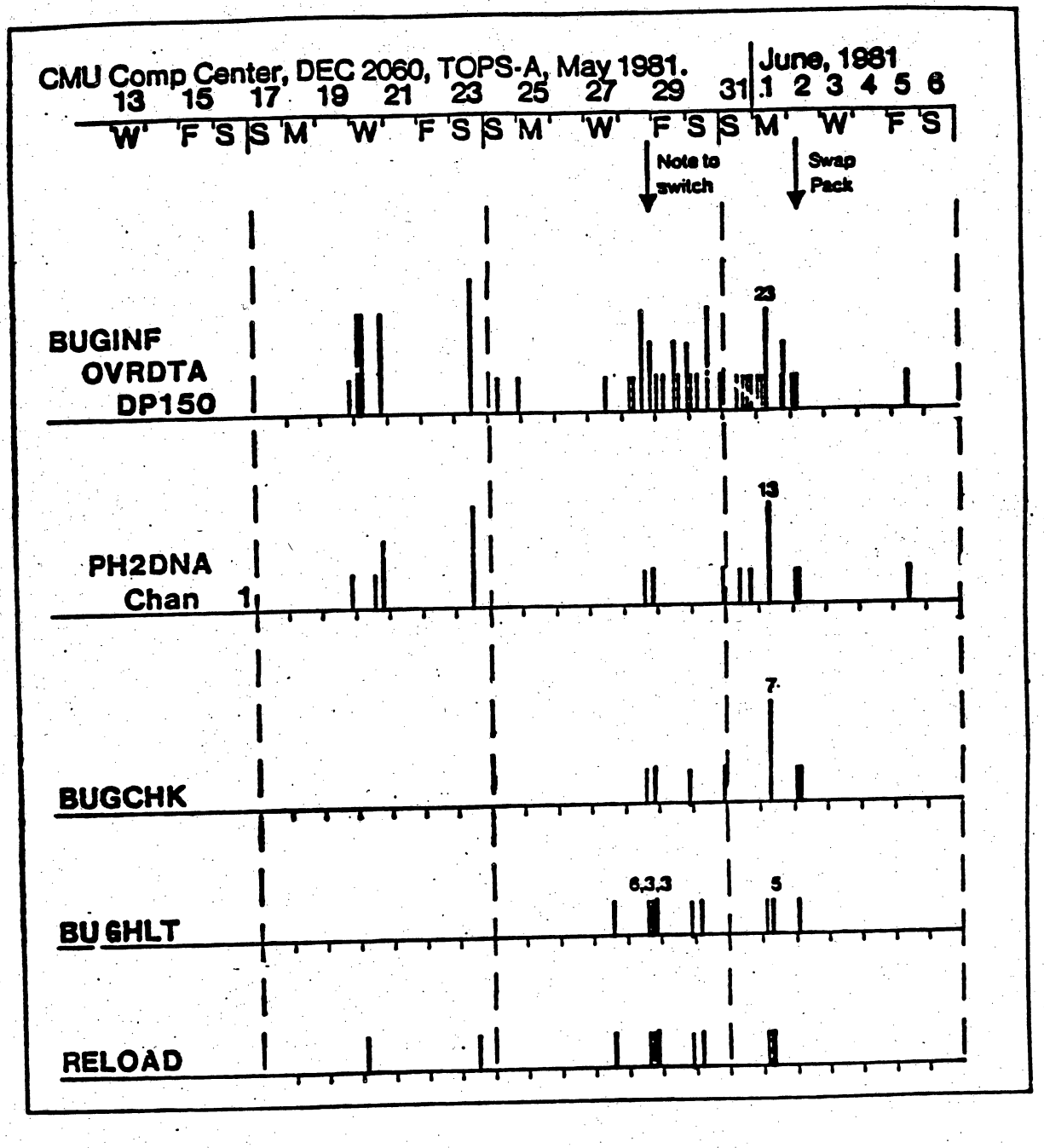
Finding Patterns

- Syntatic feature
- Semantic features
- Temporal features

Trend Analysis

Clustering

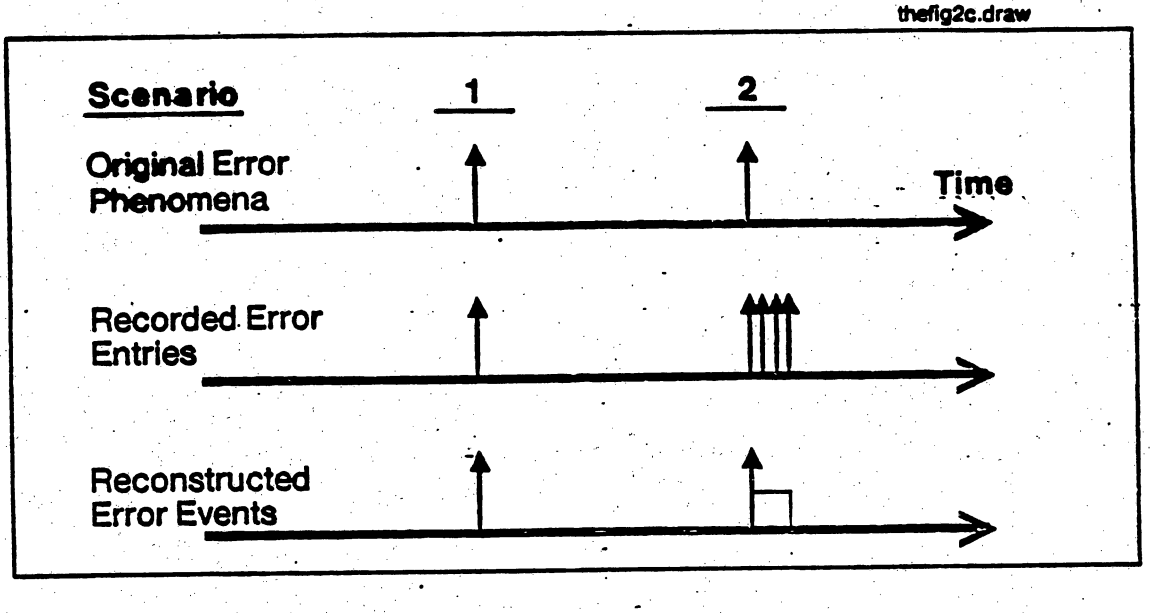
Manually Derived Trend Analysis
- RP06 on TOPS-A.



Example of a System Error File

```
SEQ    TIME    Sat 23 May 81
-----------------------------------------------------------------------
11. 10:28:57 LP20 STD STAT=146 GEN STAT=0
-----------------------------------------------------------------------
12. 11:07:03 DL11C CONFIGURATION REPORT
13. 11:07:03 DH11 CONFIGURATION REPORT
14. 11:07:03 LP20 CONFIGURATION REPORT
15. 11:07:04 CD20 CONFIGURATION REPORT
16. 11:07:04 DM11BB CONFIGURATION REPORT
17. 11:07:04 DL11C CONFIGURATION REPORT
18. 11:07:04 DH11 CONFIGURATION REPORT
19. 11:07:04 LP20 CONFIGURATION REPORT
20. 11:07:04 CD20 CONFIGURATION REPORT
21. 11:07:04 DM11BB CONFIGURATION REPORT
22. 11:07:04 RELOAD OF Carnegie-Mellon University TOPS-A, TOPS-20
                    Monitor 4(3354)-2 VERSION 4(3354)-2
                    BUILT ON Mon 27 Apr 81 13:44:42 REASON PS BROKEN
-----------------------------------------------------------------------
23. 11:18:38 DL11C CONFIGURATION REPORT
24. 11:18:38 DH11 CONFIGURATION REPORT
25. 11:18:39 LP20 CONFIGURATION REPORT
26. 11:18:39 CD20 CONFIGURATION REPORT
27. 11:18:39 DM11BB CONFIGURATION REPORT
28. 11:18:39 DL11C CONFIGURATION REPORT
29. 11:18:39 DH11 CONFIGURATION REPORT
30. 11:18:39 LP20 CONFIGURATION REPORT
31. 11:18:40 CD20 CONFIGURATION REPORT
32. 11:18:40 DM11BB CONFIGURATION REPORT
33. 11:18:40 RELOAD OF Carnegie-Mellon University TOPS-A, TOPS-20
                    Monitor 4(3354)-2 VERSION 4(3354)-2
                    BUILT ON Mon 27 Apr 81 13:44:42 REASON SUBSYS BROKEN
34. 11:18:40 BUGCHK DIRFDB AT Sat 23 May 81 11:16:59 USER OPERATOR
                    CONI APR= 7740,3 CONI PAG= 0,660123 ERA= 202000,5714
35. 11:18:43 BUGCHK DIRFDB AT Sat 23 May 81 11:16:59 USER OPERATOR
                    CONI APR= 7740,3 CONI PAG= 0,660123 ERA= 202000,5714
36. 11:[illegible]:20 NCU STARTED PROGRAM: NETCON VER:4(25)-2
                    STARTED ON NODE TOPSANETCON
-----------------------------------------------------------------------
37. 11:23:23 DOWN-LINE LOAD OF NODE DN20A BY NODE TOPSA
                    LINE DESIGNATION DTE20_1_0
                    FILE LOADED PS:<SUBSYS>DN20A.SYS.12
38. 11:24:36 DETACH TU72 S/N:21287 AS MTA0 AT CHANNEL #3 CONTROLLER #0
                    UNIT #0 REASON: Xero xrun
-----------------------------------------------------------------------
```

Deducing Original Events



Related Concepts

Error Record or Error Entry

Each reported error is entered into the system error file.

Clustering Algorithm

The heuristics used to group or coalesce error records in a system error file into logical groupings.

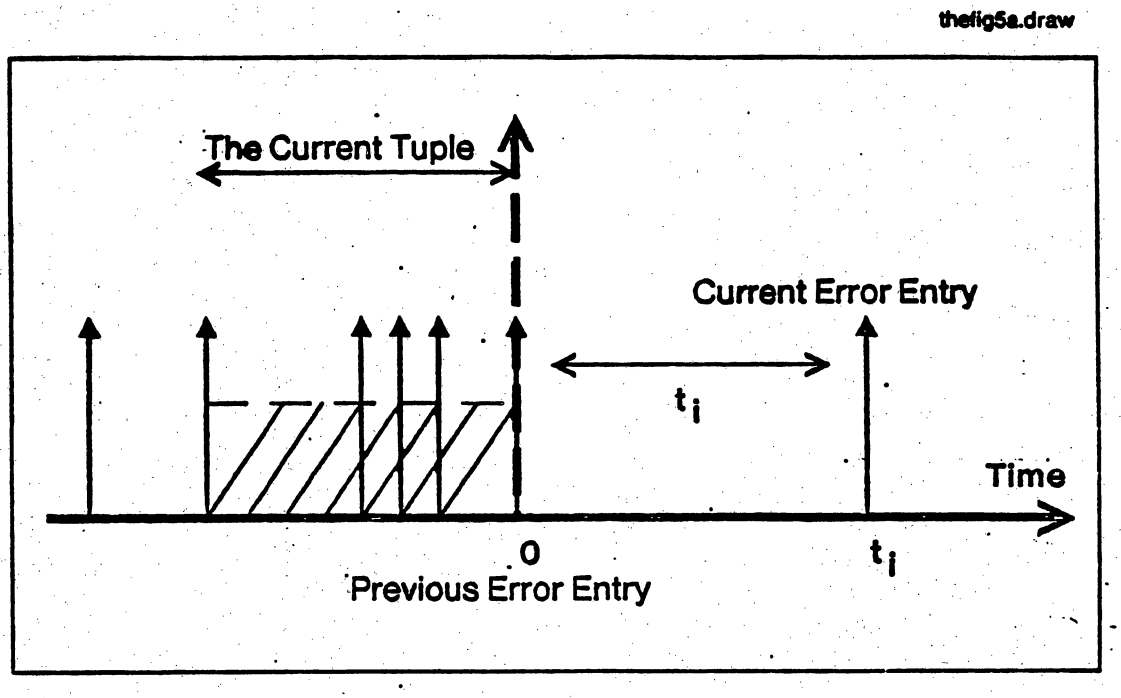
Attributes of a Tuple

Parameters that describe a Tuple.

Hierarchy of Attributes

The attributes can be organized into a hierarchy.

Tuple Clustering Algorithms



VALIDITY OF TUPLE ALGORITHMS

330 Machine-days - - - - - 58,305 Entries

Two Procedures

- Cluster time = 2.8 minutes
- Cluster time = 22.5 minutes
- No physical location information
- Same entry type as any current entry

TUPLE RESULTS

95% attributable to procedure 1

Factor of 20 reduction

- 58,305 entries ---> 3,050 entries
- 55,245 entries ---> 2,452 tuples

Five categories of tuple types

- Normal status reports
- Normal (soft) device errors
- Recoverable system crashes
- Known hard failures
- Unidentified

Categories of Tuple Types

1. **Normal operational status report** entries.
   (Such as Seq. 11.)

2. **Normal device errors**, usually self recovered, nuisance entries.
   (Seq. 38, detaching tape drives.)

3. **Recoverable system crash**, a major interruption of system service, but usually recoverable.
   (Seq. 12 to 22, Seq. 23 to 36.)

4. **Known hard failure symptom**, be it software or hardware caused.
   (Such as the OVRDTA and PH2DNA pair.)

5. **Totally unexpected**, or of such a rarity, that only a few engineers know about it.

TUPLE EXTRACTION

Concept of Tuples

Hierarchy of Attributes

- Starting time
- Spanning time
- Unique entries

- Event type sequence and latency
- Number of events of particular type

- Physical device location
- Logical structure
- Serial number
- Address

- Device error registers

MATCHING

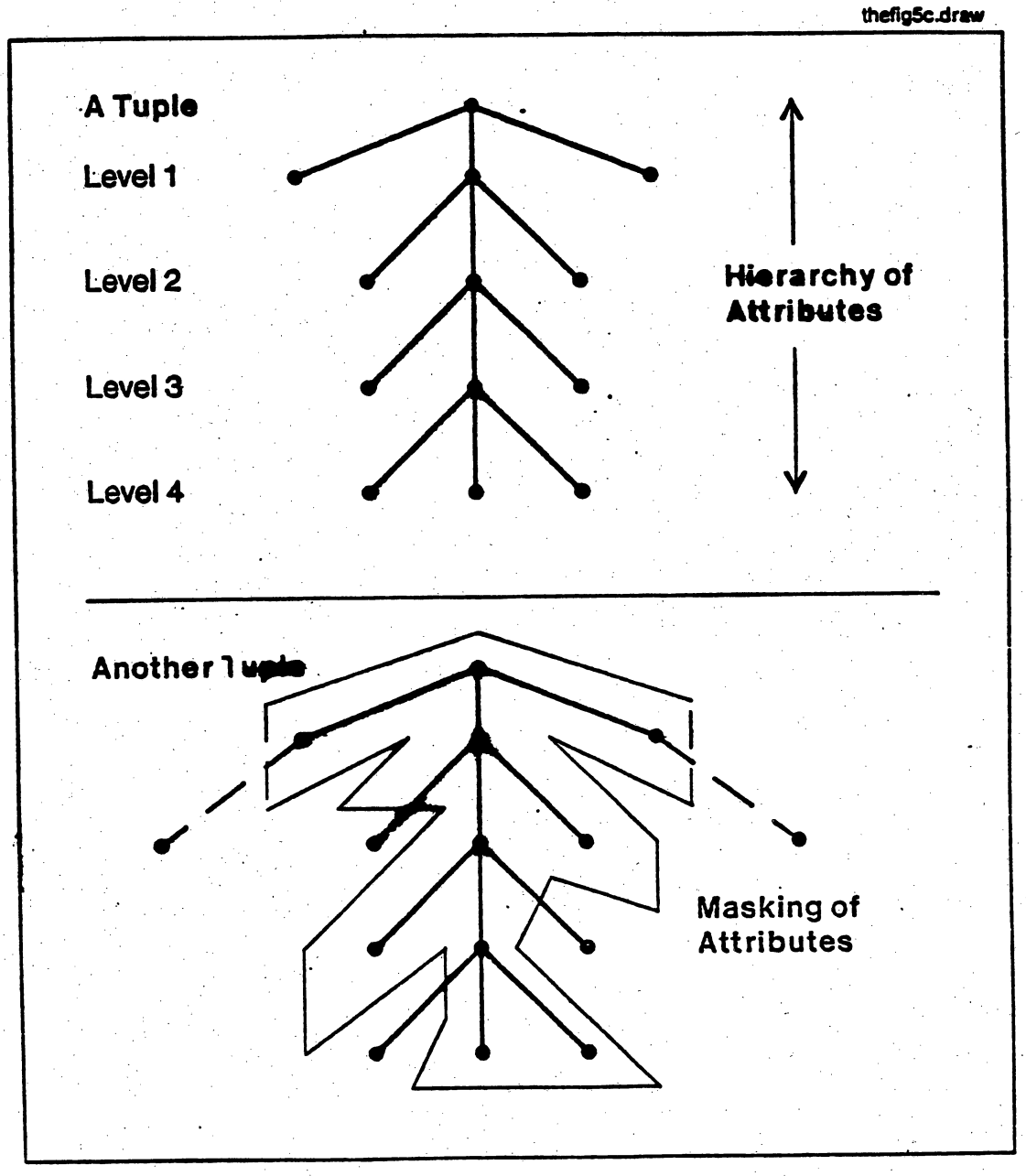
Determine uniqueness of a tuple.

Matching techniques, increasing rigor.

- Time
  - Spanning time
  - Entry count
  - Tuple degree size
- Entry type
  - Entry count
- Device specific information
  - Logical structure
  - Serial number
  - Address
- Error registers
- Individual error bits

Tuple Types

and Matching Algorithms



MATCHING RESULTS

Factor of 10 decrease in information

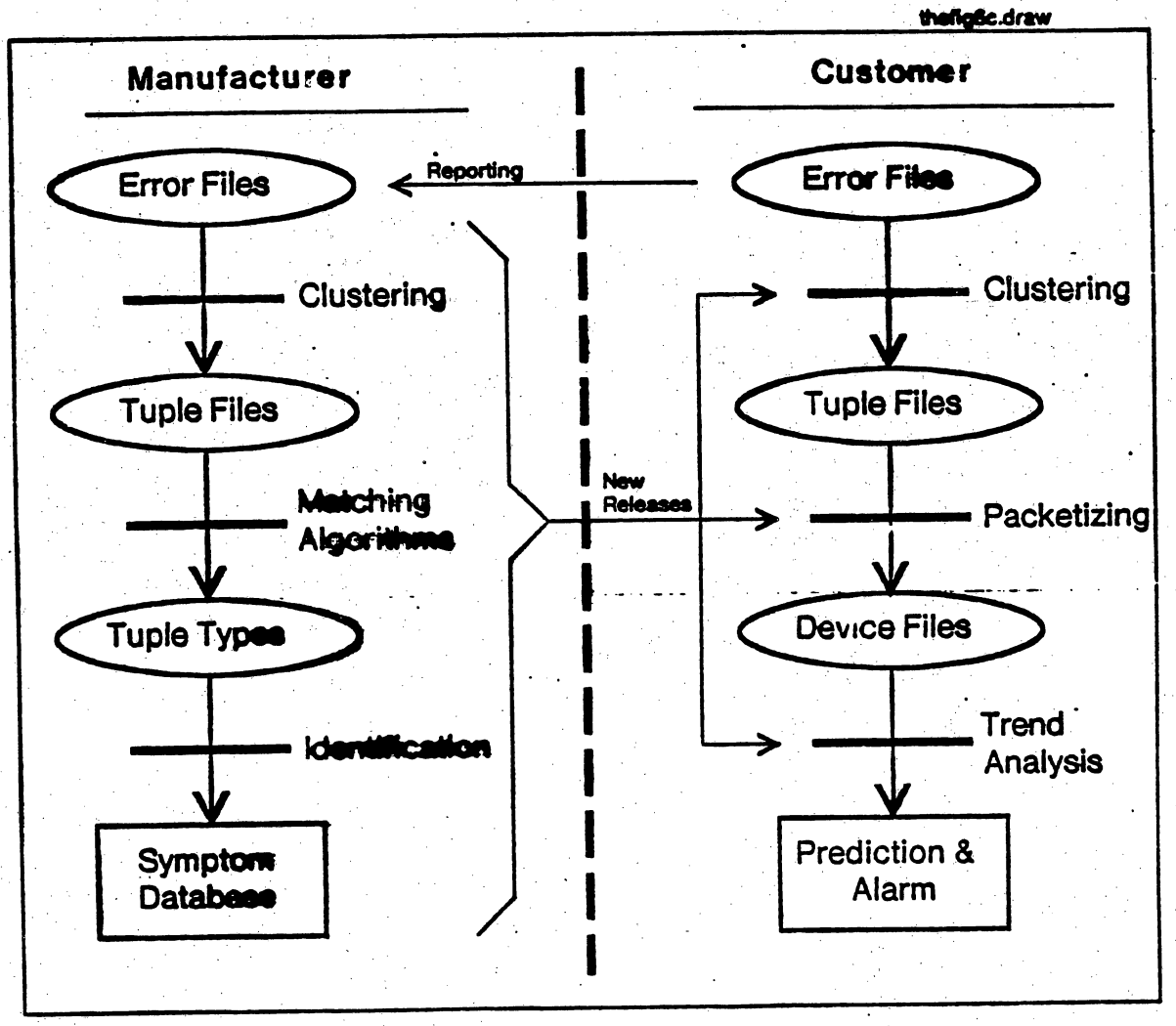
2,452 tuples ---> 255 types

4 types = 65.2%

75% of unique types in 15% of observation period

Reduce system event log from 200 ---> 2

Application of the Tuple Concept



DISTRIBUTED DIAGNOSTIC PROBLEM SOLVING

Network of hosts

Diagnostic server

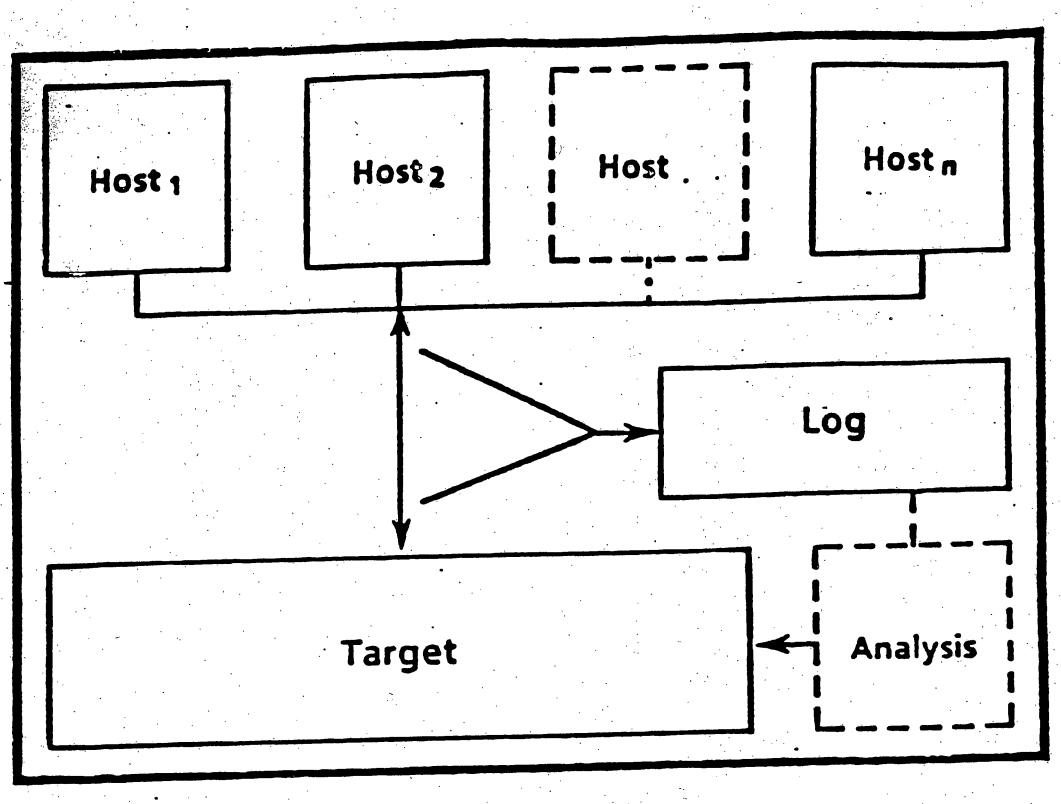
Event logging

Model-based diagnosis

Trend analysis

Fault prediction

Maintenance tool


Host 1
Host 2
Host .
Host n
Log
Target
Analysis

FAULT-TOLERANT HUMAN-MACHINE INTERFACE

Limited, but complete functionality

Focused direction of procedures

Location of user in procedure space

Consistency of commands and responses

Concurrent, context-sensitive help

Additional help on demand

Iterative development

Appeal to common sense

6.) Commercial Architectures

Goals Determine System Design Philosophy

1. Reliability vs. Availability
2. Grain of
    - Correct Data Output
    - Loss of Data
3. Transparency to User
4. Dedicated or General Purpose
5. New Design or Add On
    - Penalty to Non-Reliable Users
6. Trade Performance for Reliability
7. Stand-alone vs. Multiple Processors

Application Taxonomy

- High Availability. Occasional loss of single user but not system. General purpose. Low coverage. Tandem Stratus.
- Long Life. No manual maintenance. Automatically detect and reconfigure. High coverage. Voyager
- Postponed Maintenance. Survive failure until periodic maintenance. Aircraft
- High-Performance Computing. Susceptible to transients and hard failures due to complexity. Cray
- Critical Computations. Usually real time control systems. High coverage and low recovery time. Flight Control, Industrial Process Control (August Systems).

Fault Tolerant Computers

1944 Bell Relay Computers
Biquinary Code
Completion of Instruction

1951 Univac I
Parity
Duplicate ALUs

1957 SAGE
Duplicate Processors

1963 OAO
Quad Components

1964 ESS #1
Duplication
Coding
Watchdog Timer

1964 S/360
Scan Out
Diagnose Instruction

1967 Saturn V
TMR

1969 STAR
Coding
Standby Spares
TARP

1972 S/370
SEC/DED Code
Instruction Retry
Checkpoint/Restart
Microdiagnostics

1972 Plessey System 250
Multiprocessor

1973 Pluribus
Multiprocessor

1973 Boeing 747

1975	Tandem 16 Non Stop Multiprocessor
1976	Space Shuttle 5 Processors
1976	Amdahl 470
1979	SIFT

Table II–3. Evolution of IBM maintenance strategy.

Machine	*Era*	*Techniques*
650	Late 1950s	Six internal checkers
		Stand-alone diagnostics on punched cards
		Light and switch panel
1401	Early 1960s	20 internal checkers
		Stand-alone diagnostics
		Light and switch panel
S/360-50	Mid-1960s	75 internal checkers
		OLTEP—On-Line Test Executive Program
		Microdiagnostics
		Log fault data to main memory. EREP—Error Recording and Edit Program for outputting logged data
		Maintenance panel
370/168 Mod 3	Early 1970s	Error-detection circuits
		OLTEP
		Microdiagnostics for fault isolation
		Service processor, including trace unit—trace up to 199 fixed and 8 movable logic points over 32 machine cycles for intermittent or environmental faults
303X	Mid-1970s	Error-detection circuits
		OLTS—On-Line Functional Tests
		Console and processor microdiagnostics
		EREP
		Scope loops
		Support processor, including trace and remote (telephone) access to log data and trace information
4341	Late 1970s	Error-detection circuits
		25,000 shadow latches
		Support processor—error logging and environmental monitoring

Reliability, availability, and serviceability for IBM 4300's

Error-detection/correction circuits

- Single-bit error correction/double-bit error detection in main memory
- Data-path parity
- Store-and-fetch memory access protection
- Instruction retry (model 4341 only). On an error, processor performs a retry and, if successful, loads the machine check interrupt. Options include hard-stop on error, no retry (but logout), disable error report, and stop after logout. During instruction execution, data in certain machine facilities is saved. Prior to instruction re-execution, this data is restored.
- Disk error correction
- I/O retry at both processor and disk controller levels
- Peripheral unit "power off" signal (model 4341 only)
- Disk self-test
- Voltage margin under program control
- Relocatable control storage
- Level-sensitive scan design
- Microdiagnostic locate, field-replaceable unit 80 percent of time
- Halt or trace on address or data match comparisons can be made on any reference, I/O reference, data store, or instruction fetch

Support processor

- Separately powered
- Separate system diskette for microcode loading, system error logging, and storage of microdiagnostics
- Sensors for monitoring power variances, temperature fluctuations, and electrostatic discharge (model 4341 only)
- For both retriable and unretriable errors, the support processor performs an internal logout. Each logout has an identifier that specifies the number of logouts to date.
- Support-processor-generated eight-digit reference code guide to failing unit. Reference code logged on diskette and display console.
- Display console and data link functions for remote support facility

Remote support facility

- Remote monitoring (especially of error registers) and control
- Remote initiation of diagnostics
- Remote examination of error log on system diskette
- Distribute microcode updates

Table II–5. Error-detection hardware in the UNIVAC 1100/60.

Memory
Double-error-detecting code on memory data
Parity on address and control information
Cache.
Parity on data. address. and control information
I/O Unit
Parity on data and control
CPU
Parity on data paths
Parity on control store
Duplication and comparison of control logic

Table II–6. Error recovery in the UNIVAC 1100/60.

Memory
Single-error-correction code on data
Retry on address or control information parity error
Cache
Retry on address or control information parity error
Disable portions of cache on data parity errors
I/O unit
Retry on data or control parity errors
CPU
Retry on control store parity error
Invert sense of control store
Macroinstruction retry

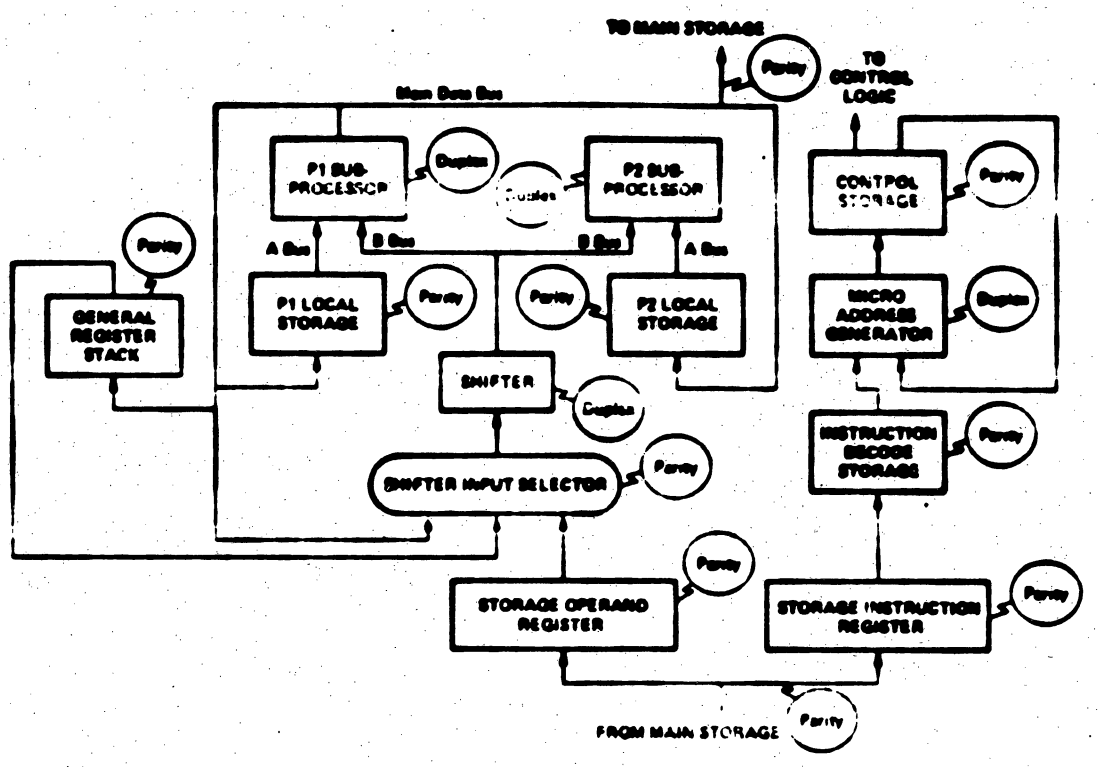


Figure 2. Block Diagram of the 1100/60 Microexecution Section, Showing Sample Applications of Fault Detection Techniques

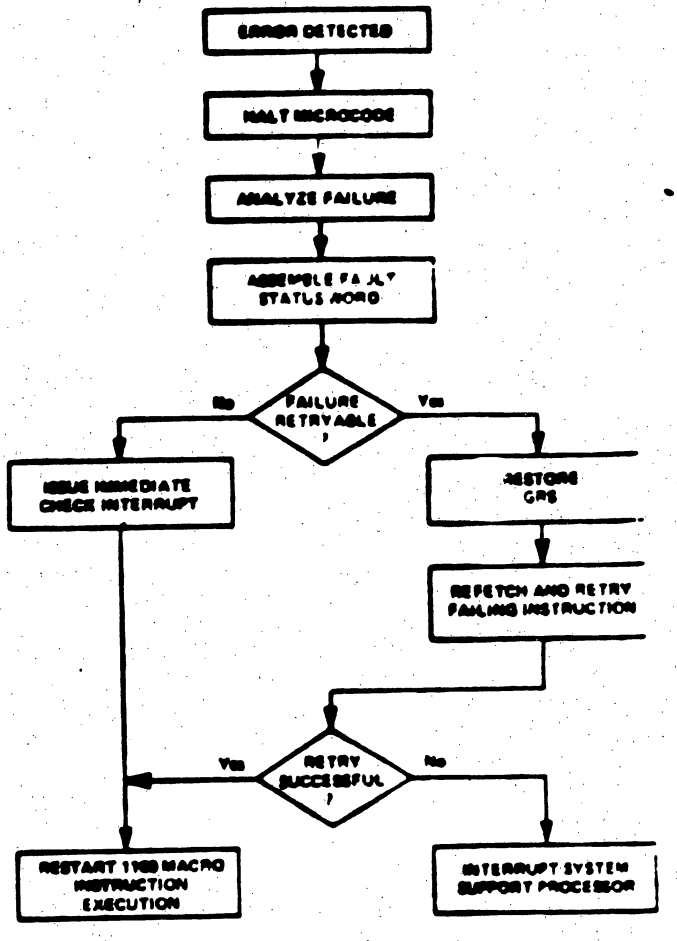


Figure 4. Flowchart of CPU Error Recovery

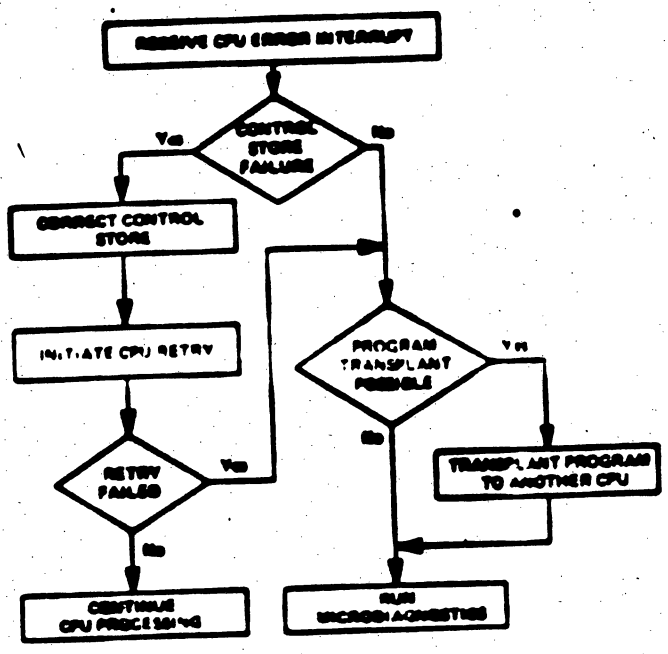
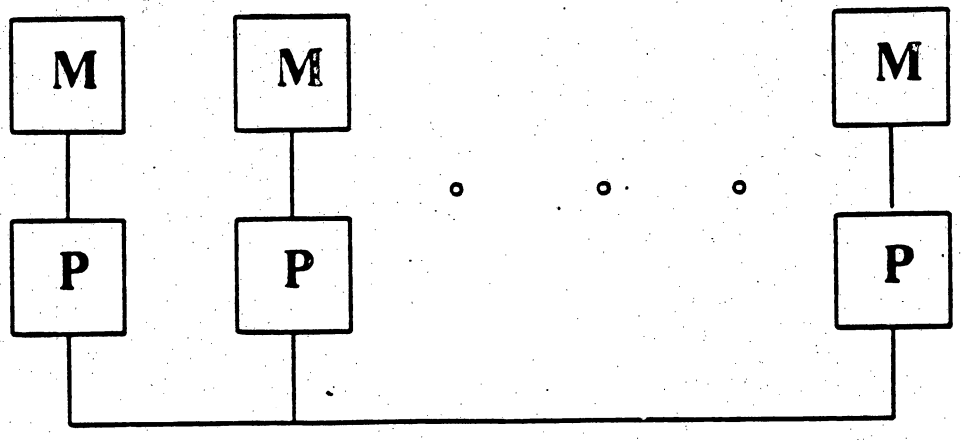


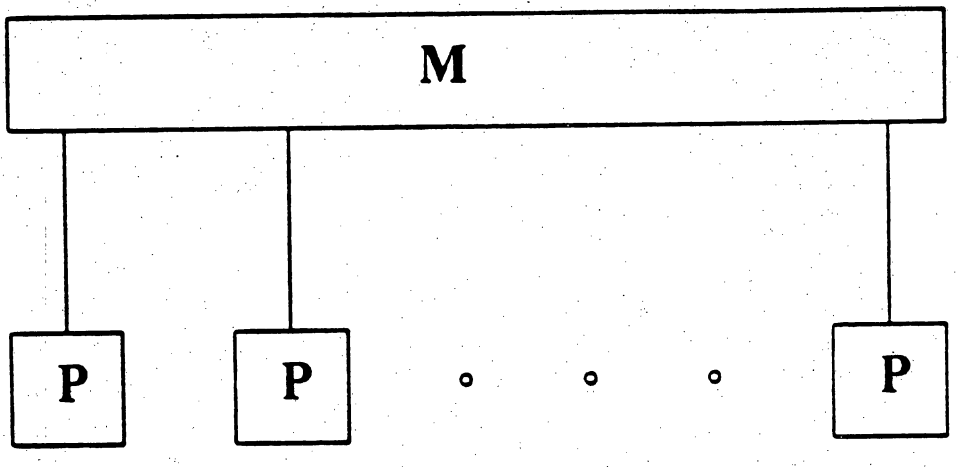
Figure 5. Flowchart of SSP Recovery Procedure

Table II–7. Uses of support processors.

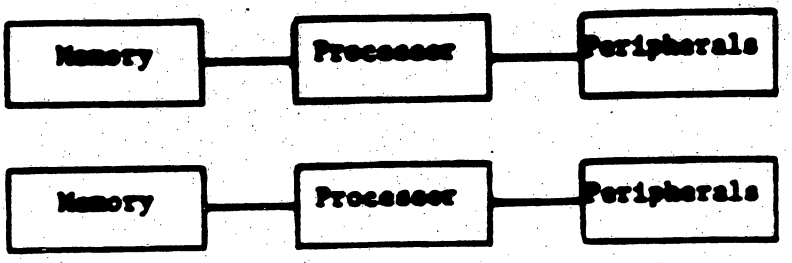
- System console
- System boot
- System quick test of boot path
- Error logger
- Diagnostic tool
  - Microdiagnostics
  - Scan/set/compare internal state
  - Fault injection
  - Remote diagnosis
- Error recovery
  - Writable control store reload
  - Transplant state to another processor
  - Reconfiguration



Multi-Computer

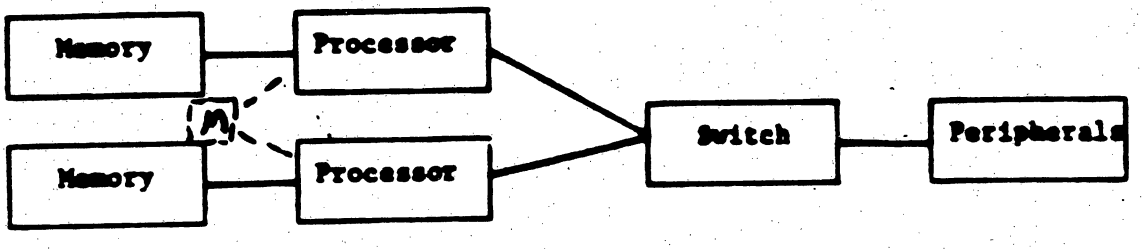


Multi-Processor

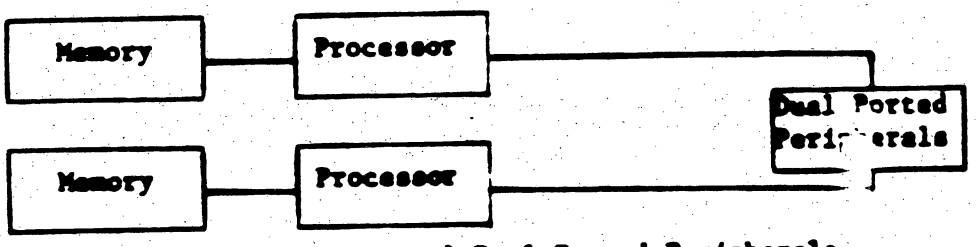


30 minute switch over

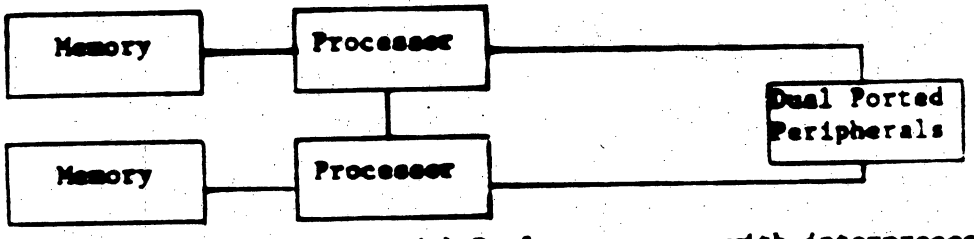
a.) Backup Computer System



b.) Backup Computer with Switch

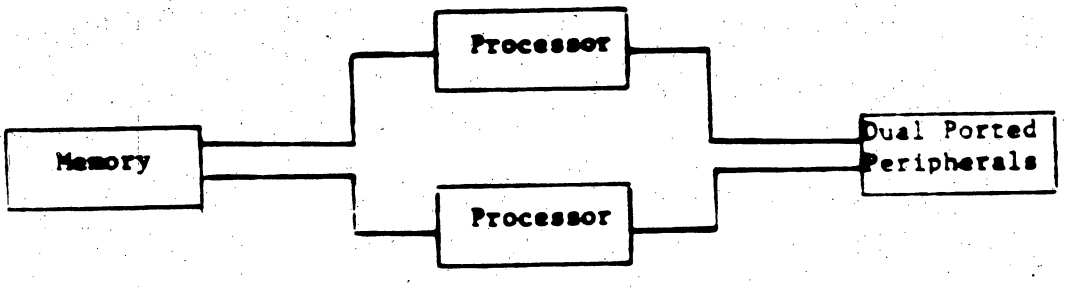


c.) Dual Ported Peripherals



Loosely Coupled

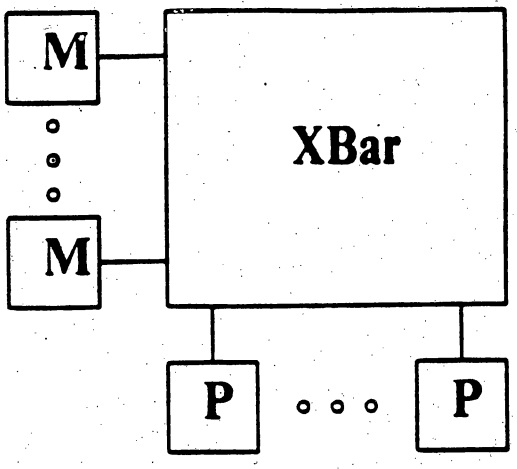
d.) Dual processors with interprocessor communication



e.) Tightly coupled multiprocessor

Figure 5: The Evolution of Systems Using Dynamic Redundancy

Lessons From C.mmp



- Self Sizing/Quick Reboot
- Self Scheduling
- Mutual Suspicion - Timeouts on Memory Ports
- Deamons
- Suspect/Monitor Mode
- Importance of Monitoring

Detection

- Diagnostic Programs
- Watchdog Timer - **Tandem, Tolerant, Synapse, ESS**

  "I'm alive"
- Self Checking Modules - **Stratus, Sequoia, Intel, ESS**

Recovery

- Forward Error Recovery
  - continue processing
    - **ESS**
- Backward Error Recovery
  - record process state
    - journaling 5%
      - **Auragen**
    - checkpoint/mirroring 20-30%
      - **Tandem, Tolerant, Synapse**
    - duplex 100%
      - **Stratus, Intel**

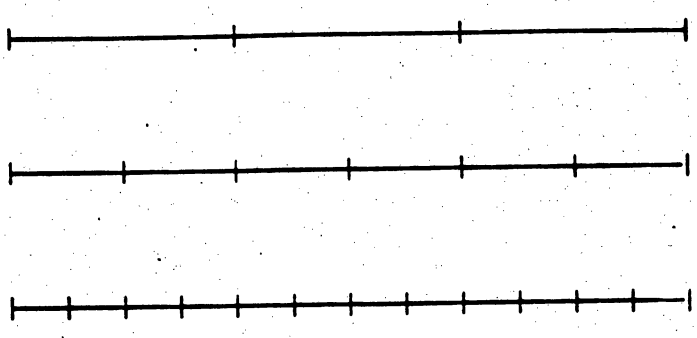
Dynamic Redundancy

HW/SW

Detection	*SW*	*HW*	*HW*
Recovery	*SW*	*SW*	*HW*
	Tandem	**Sequoia**	**Stratus**
	1.3	*2*	*4*

Impact of Application on Structure

Avionics



Frames
- **Rigid Schedule**
- **Cyclic**

On-line Transaction Processing

- **Messages**
- **Human retry**

Categories of Dynamic Redundancy

UNIPROCESSOR			
Detection	SW	HW	HW
Recovery	SW	SW	HW
Example	IBM-PC	VAX	STAR
	[standard commercial systems]		
MULTICOMPUTER			
Detection	SW	HW	HW
Recovery	SW	SW	HW
Examples	Tandem Tolerant Auragen	ESS	Stratus
MULTIPROCESSOR			
Detection	SW	HW	HW
Recovery	SW	SW	HW
Examples	Synapse Parallel	Sequoia Fluribus	Intel 432

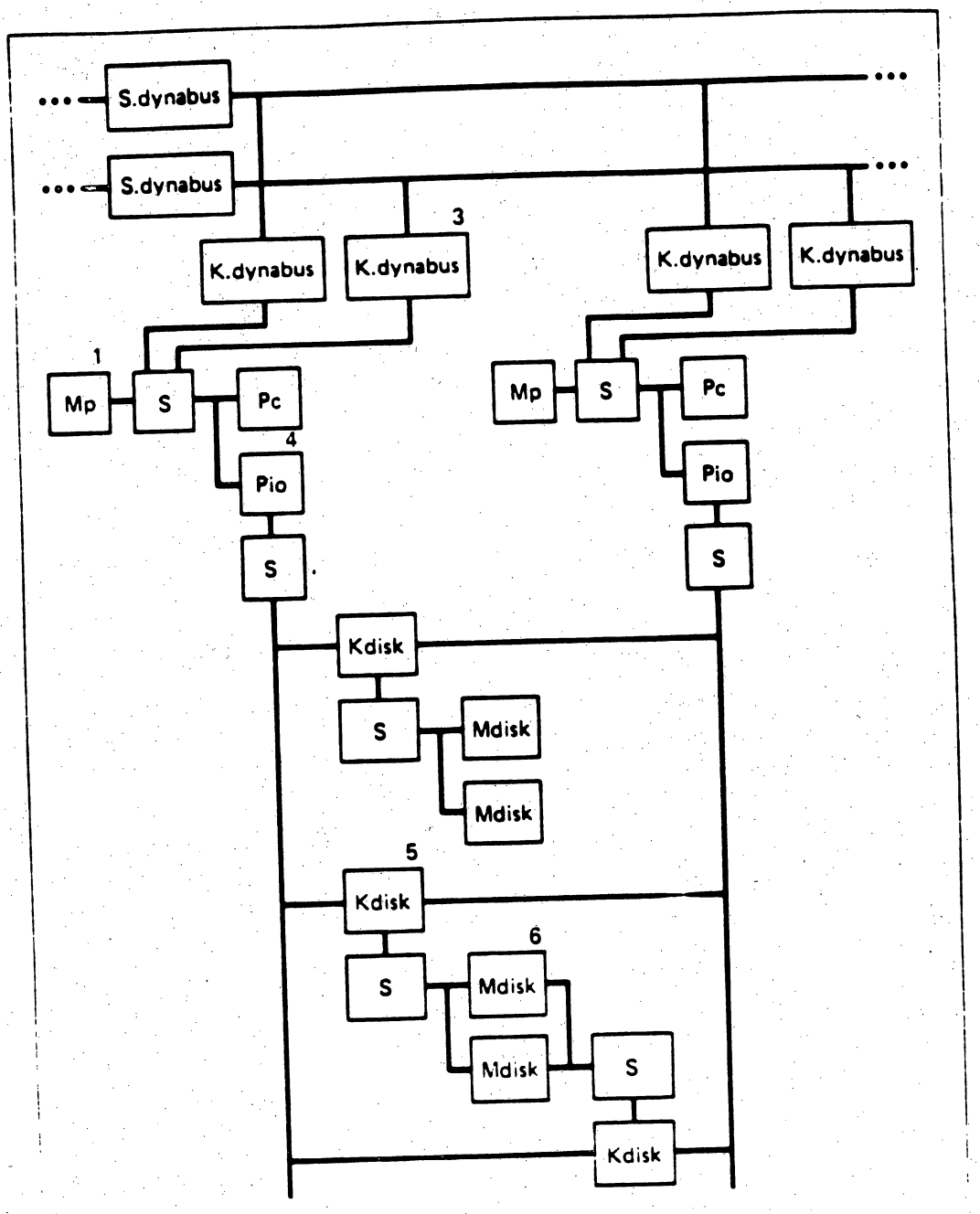
Categories of Masking Redundancy

UNIPROCESSOR		
Detection and Recovery	SW	HW
Examples	Any uniprocessor	C.vmp
MULTICOMPUTER		
Detection and Recovery	SW	HW
Examples		FTMP
	Space Shuttle SIFT	Triconex-1

SYSTEM	CHARACTERISTICS
ESS (Telephone) Dynamic Redundancy	Real Time Single Program Periphery Buffers Information High Availability
FTMP (Commercial Flight Control) Masking Redundancy	Real Time Data Redundancy Through Slow Time Variation High Reliability - correct output for critical functions
Tandem Dynamic Redundancy	Long Term Data Storage High Availability
C.vmp Masking Redundancy	Real Time Software Transparency High Reliability

TANDEM
DESIGN OBJECTIVES

- "Nonstop" operation where failures detected, components configured out of service, repaired components configured back in without stopping other system components.
- No single hardware failure can compromise data integrity of the system.
- Modular system expansion through adding more processing power, memory, and peripherals without impacting application software.


S.dynabus
S.dynabus
3
K.dynabus
K.dynabus
K.dynabus
K.dynabus
1
Mp
S
Pc
4
Pio
S
Mp
S
Pc
Pio
S
Kdisk
S
Mdisk
Mdisk
5
Kdisk
6
S
Mdisk
Mdisk
S
Kdisk

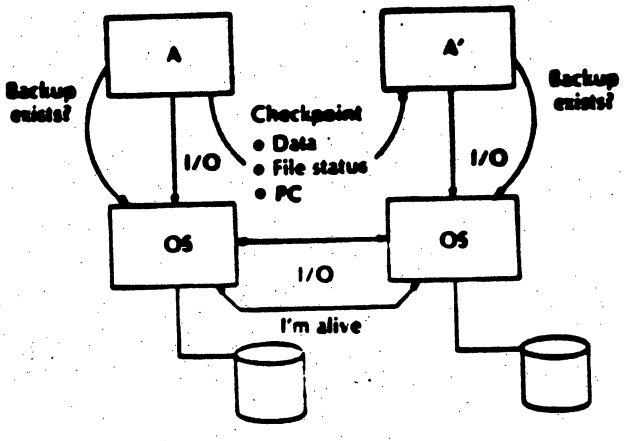


Figure II–1. Shadow processor in Tandem.

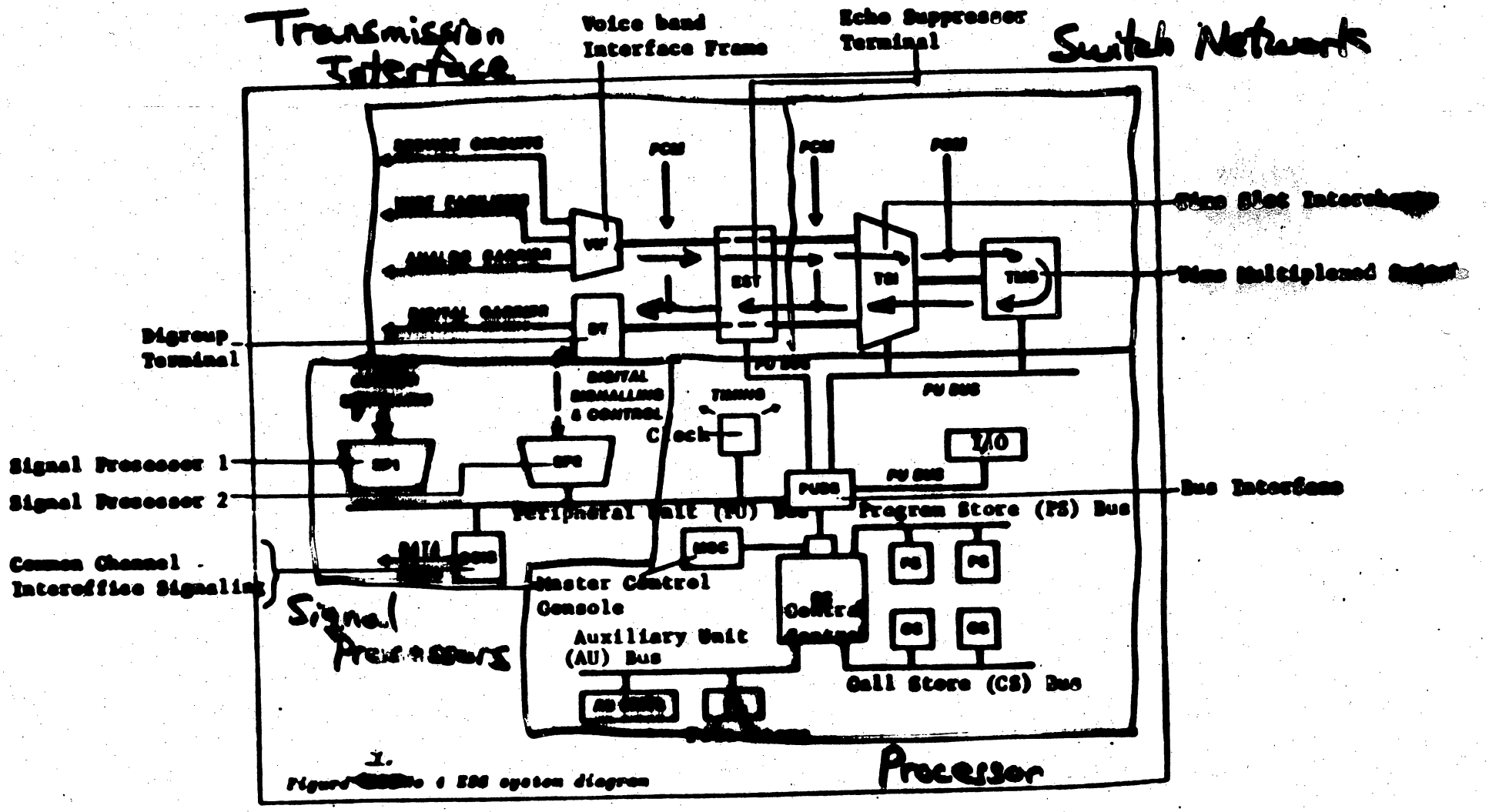
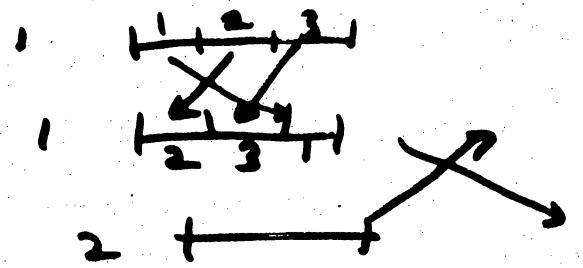


Figure 1 ESS system diagram



Phase	Recovery Action	Affect
1	Initialize transient memory	No calls lost
2	Reconfigure periphery. Initialize all transient memory.	Lose calls in process of being established
3	Verify memory operation, establish workable processor, configure periphery. Initialize all transient memory.	Lose calls in process of being established.
4	Establish workable processor, configure periphery, initialize all memory.	All calls lost

3

TABLE <XXX>

System	Number of Lines	Year Introduced	Processor	Number Installed	Comments
ESS-1	5000-65,000	1965	No. 1	1600	First processor with separate control and data memories.
ESS-2	1000-10,000	1969	No. 2	500	
ESS-1A	>10,000	1976	No. 1A		Four to eight times faster than No. 1 processor. Combined control and data store. Microcoded, emulates No. 2 processor.
ESS-2B	1000-20,000	1975	No. 3	500	
ESS-3	500-5000	1976	No. 3		

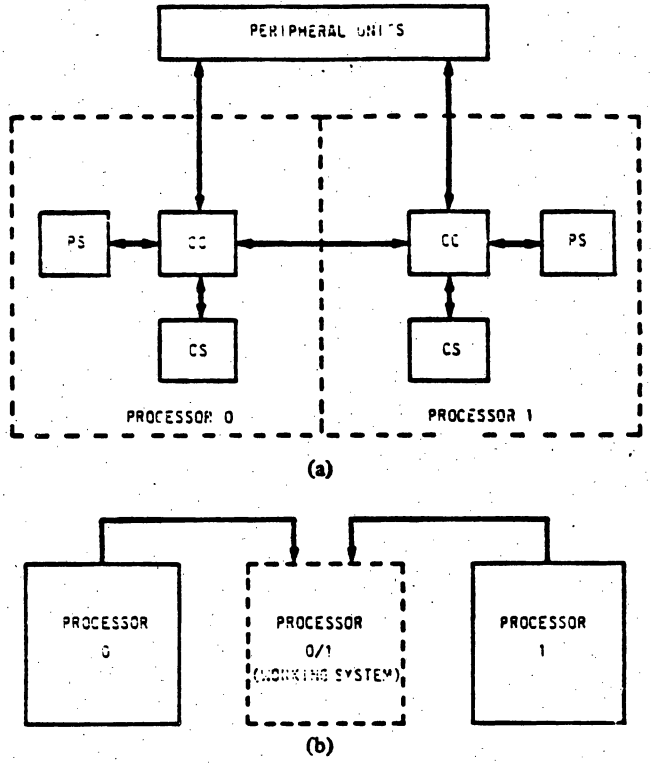


Fig. 2. Single-Unit duplex configuration. (a) Processor structure. (b) Two possible configurations.

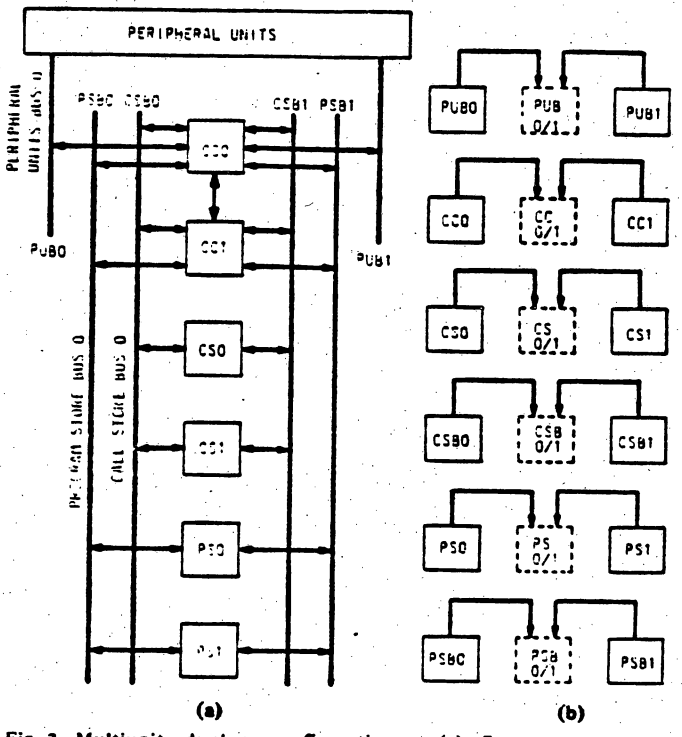


Fig. 3. Multiunit duplex configuration. (a) Processor structure. (b) 64 possible configurations.

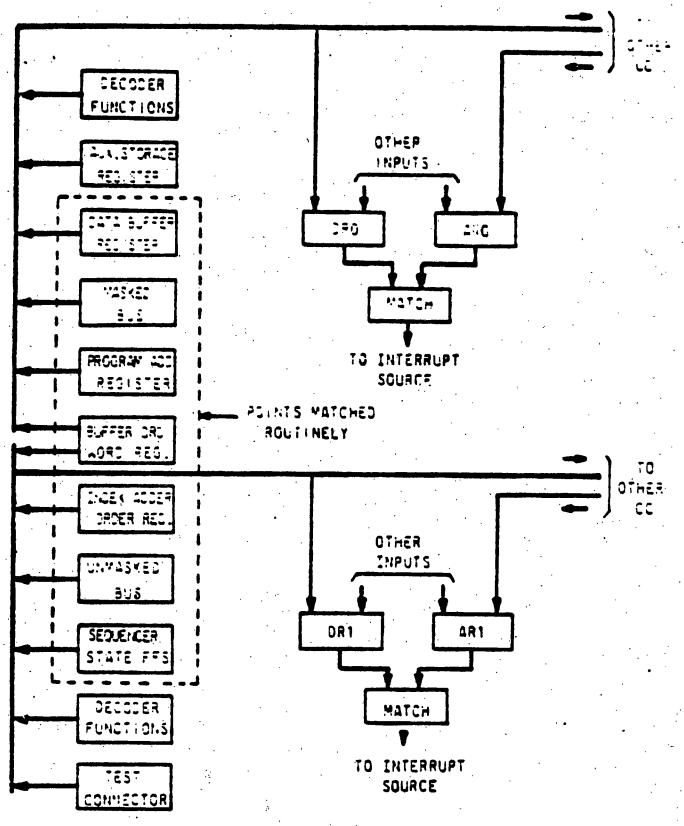


Fig. 5. No. 1 CC match access.

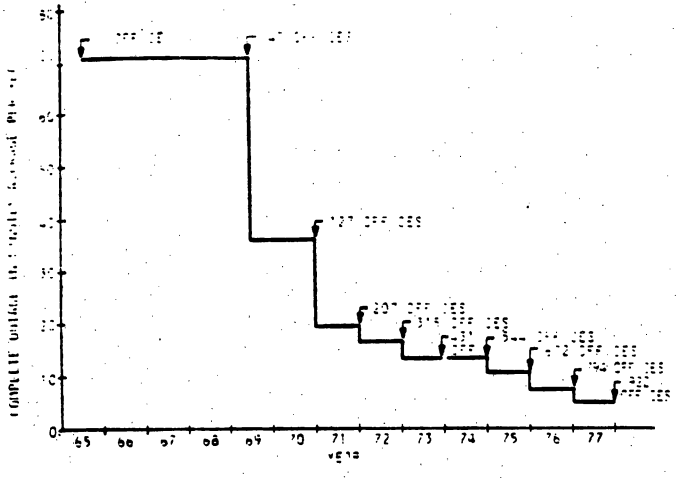


Fig. 6. No. 1 ESS Service performance.

No. 1A Processor

Bipolar SSI
4-8 times faster
than No. 1 ESS.

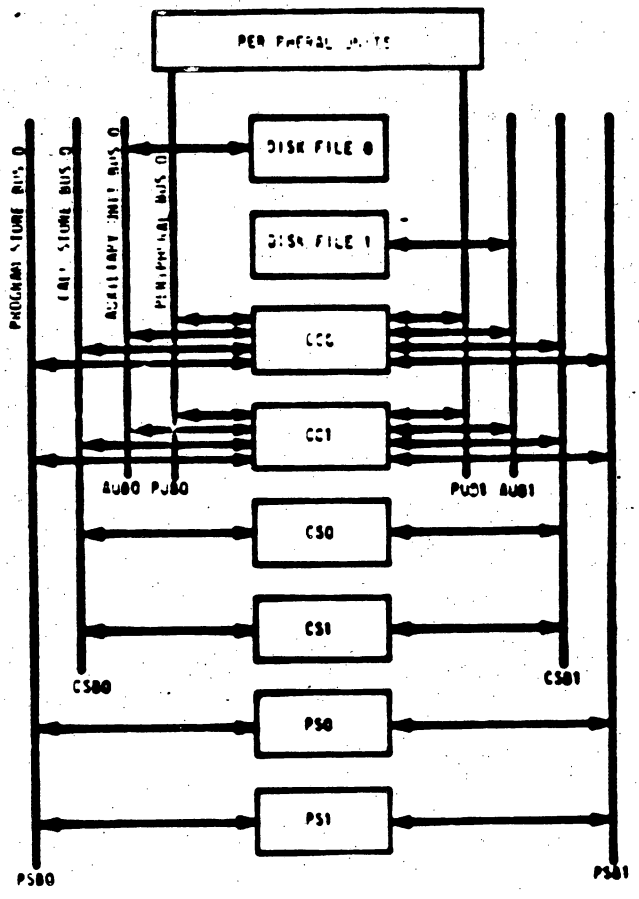


Fig. 8. No. 1A Processor configuration.

50,000 gates
checking circuits

Processor Config.
Hardware
(Timers, reset,
initialize,
reconfigure)

Parity
Duplicated

24 + 2 parity

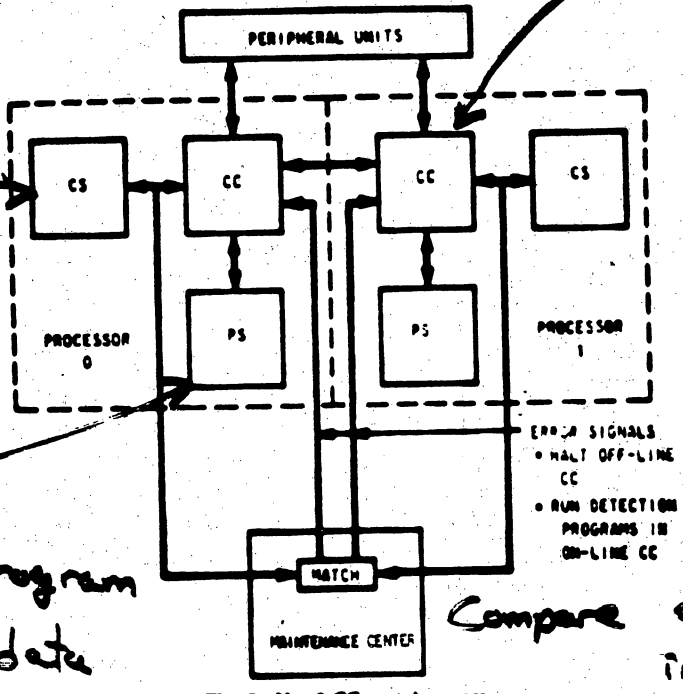
1	2	1	2	1	2

Data & address
Regenerate from
disk.

2 Match circuits
24 bits in 4 groups
matched every
machine cycle (700 nsec)
16 nodes

No. 2 ESS

5000 gates RTL



16 bits no parity

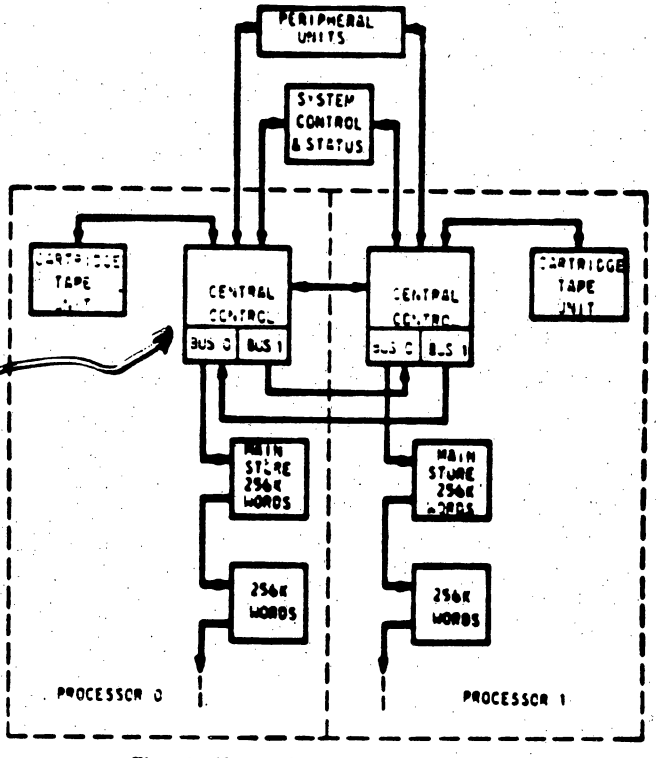
22 bits
Odd parity - program
Even parity - data

Compare call store input registers

Fig. 7. No. 2 CC match scheme.

Watchdog Timer

No. 3A



Microprogrammed

Combined CS, PS
1 μsec access

Fig. 10. No. 3A Processor organization.

No duplication/match. Use self-checking circuits for better fault diagnosis.

Standby processor halted, memory updated on writes. Parity errors on read automatically try standby memory.

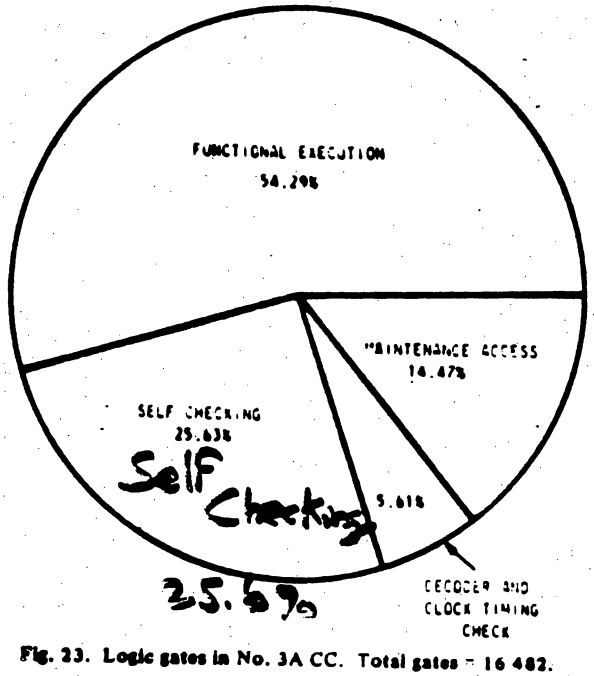
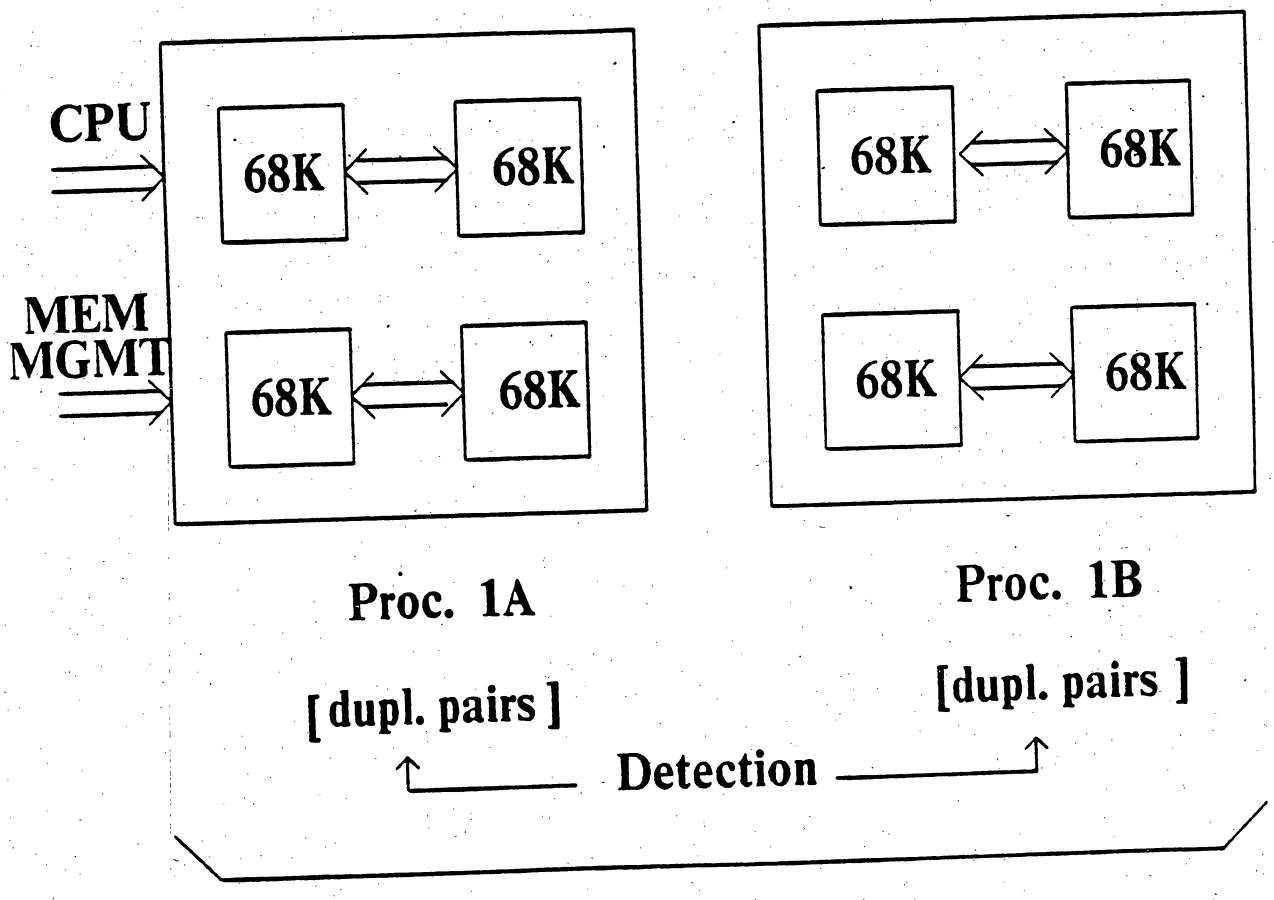


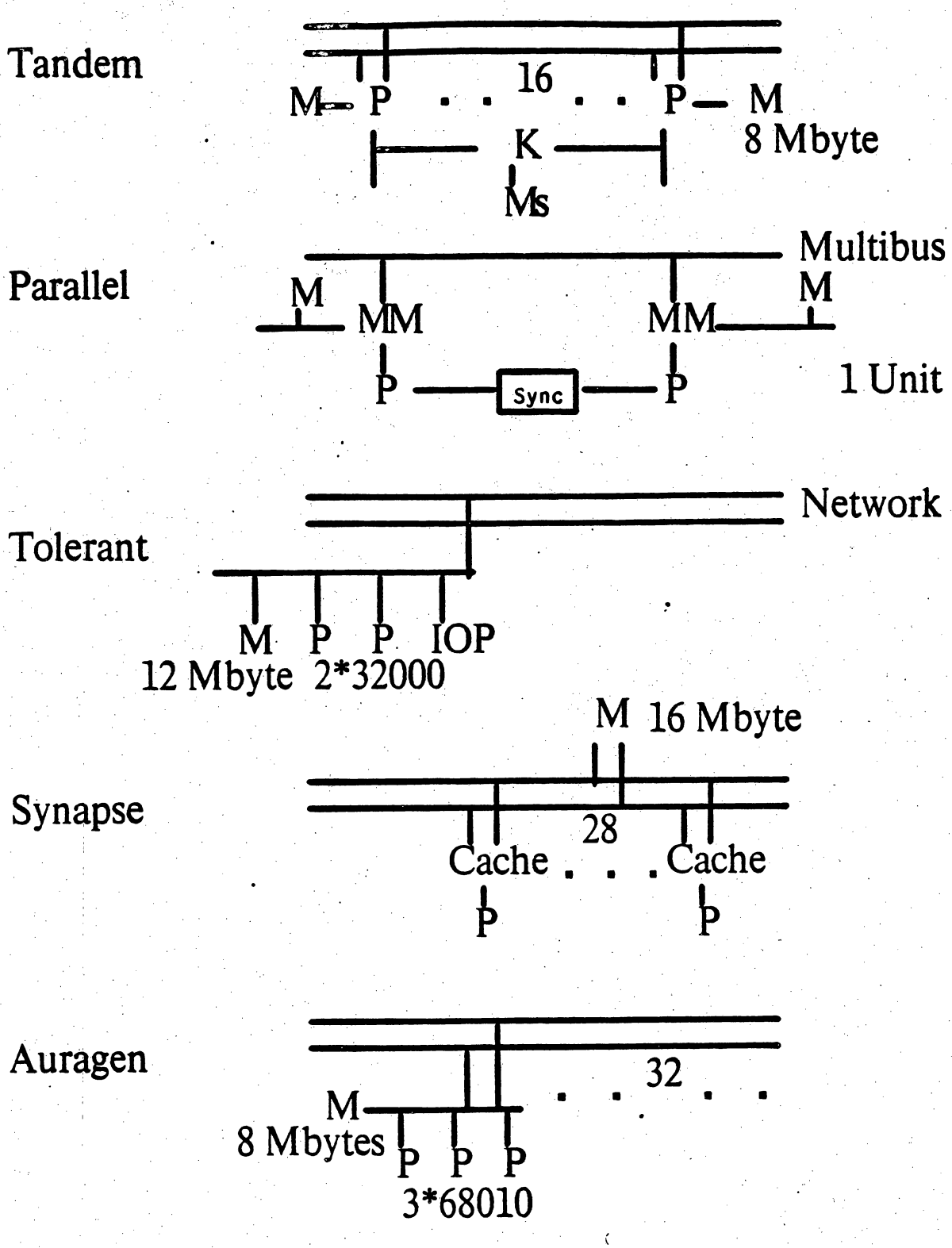
Fig. 23. Logic gates in No. 3A CC. Total gates = 16 482.

STRATUS /32

- Double Duplication
  - both for fault tolerance and fault detection
- Wind up having 8 68000's per FT processor in multi-pc. configuration.



Multiple Processor Structures



Architecture

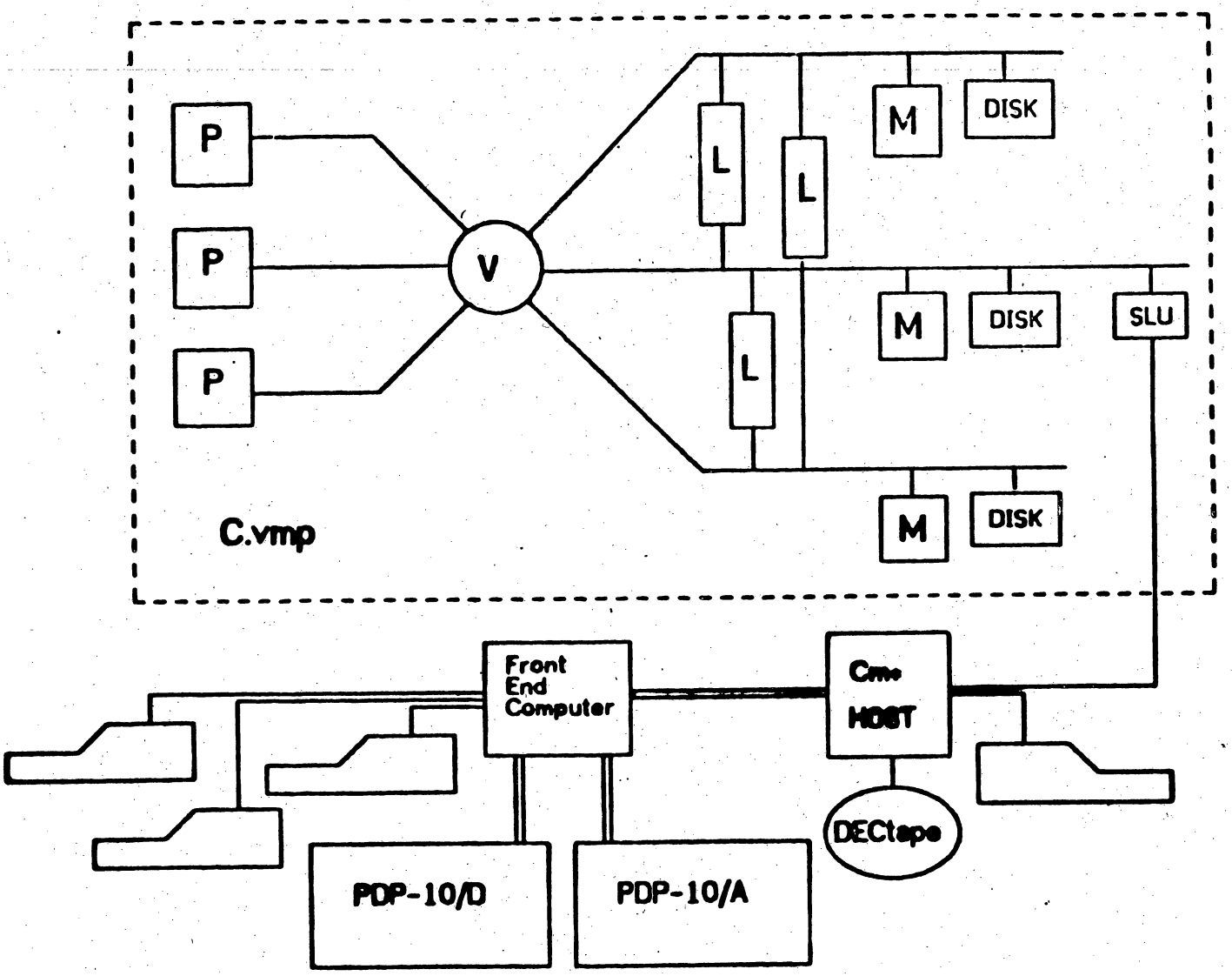
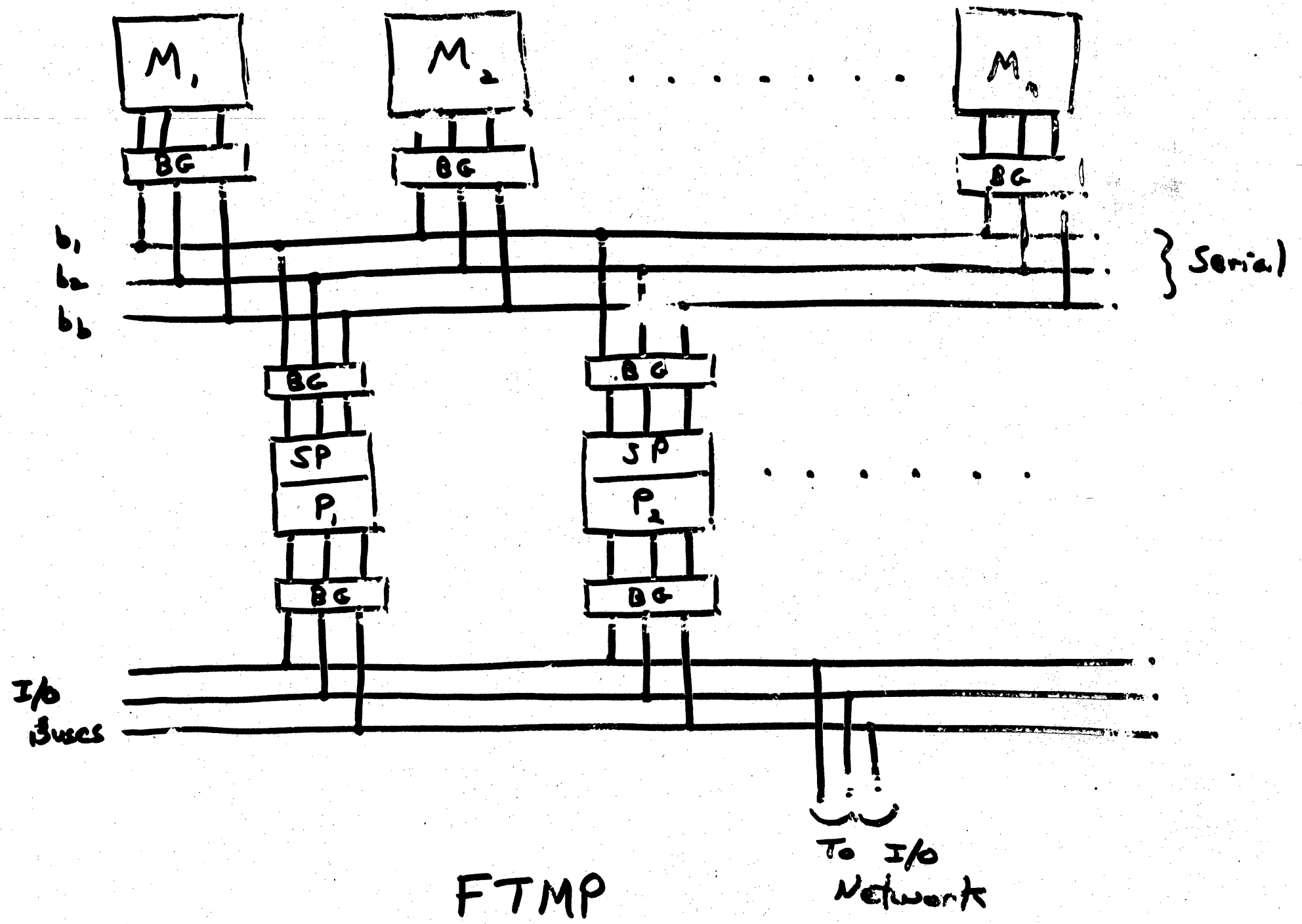
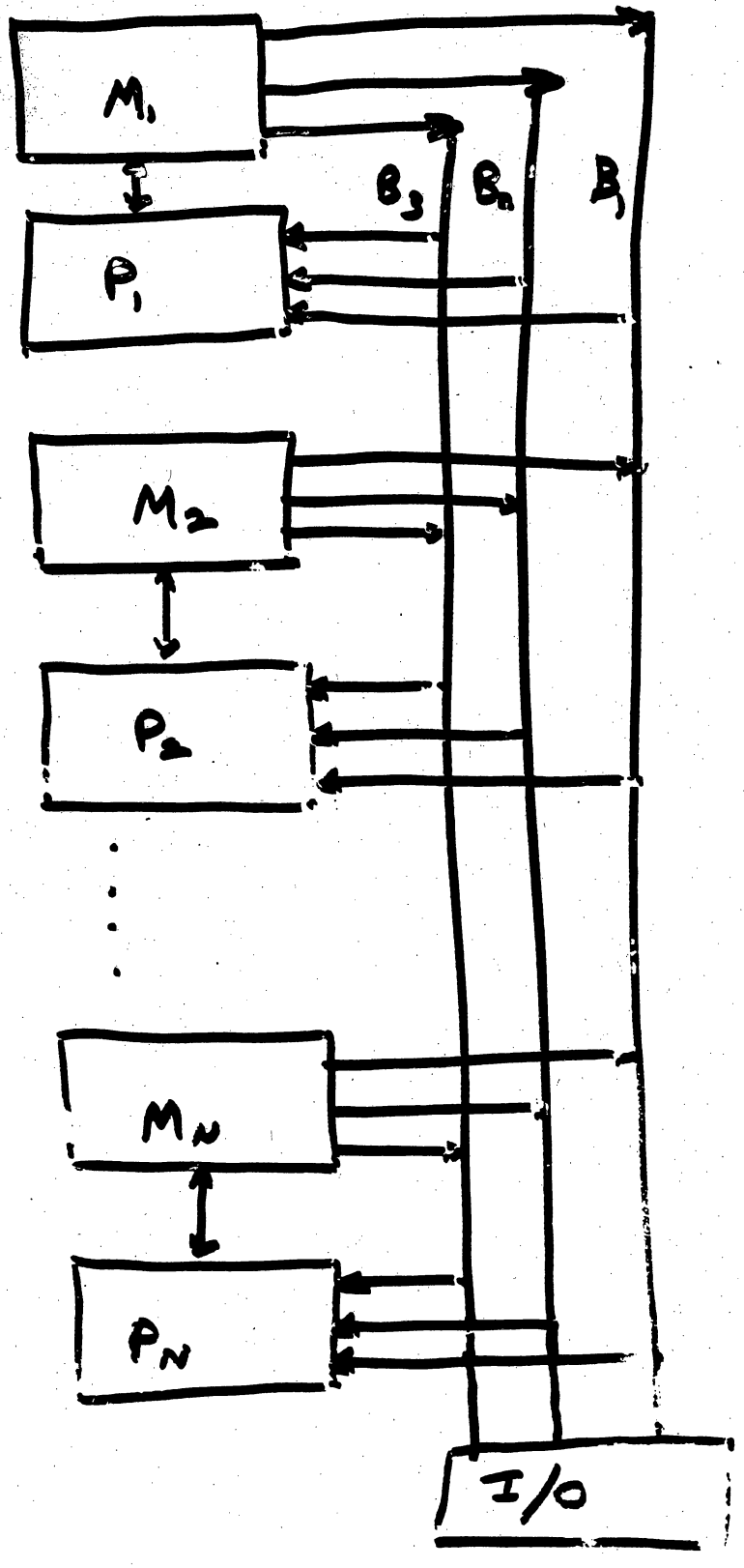


FIGURE 5.1 C.vmp CONFIGURATION AND CONNECTION TO C-MU FACILITIES



FTMP



SIFT

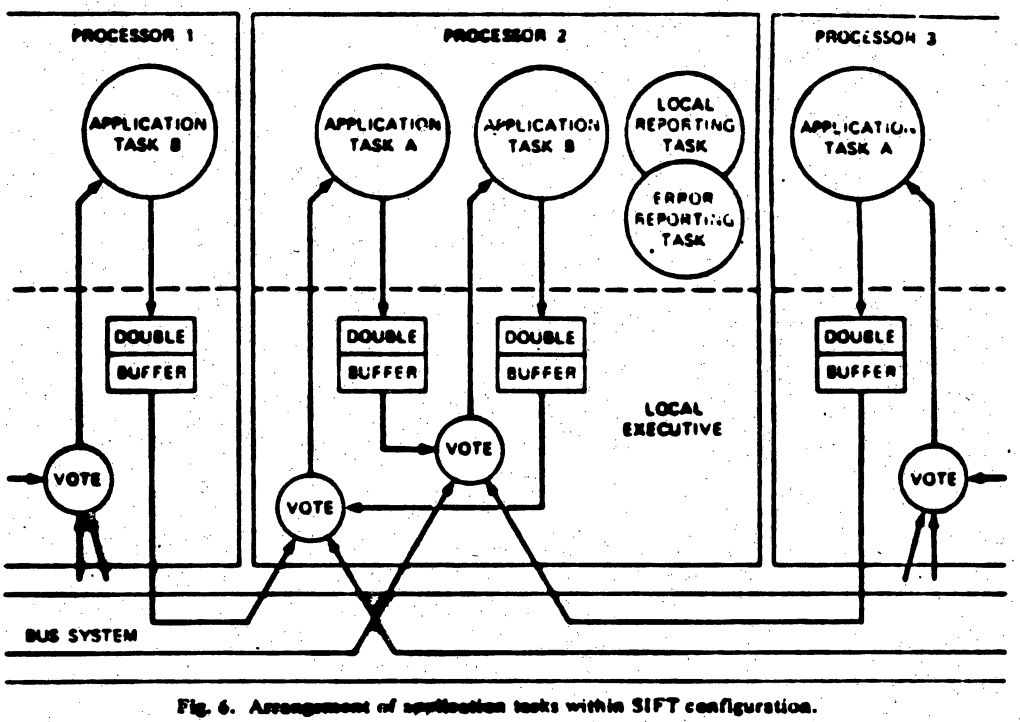


Fig. 6. Arrangement of application tasks within SIFT configuration.

FAULT TOLERANT DESIGN METHODOLOGY AND EXAMPLE:

The Intel 432

A Methodology

- Define Objectives
- Limit the Scope
- Define Confinement Regions
- Design Error Handling Mechanisms
- Design Error Reporting Mechanisms
- Testing of Error Handling/Reporting Mechanisms
- Evaluate Design

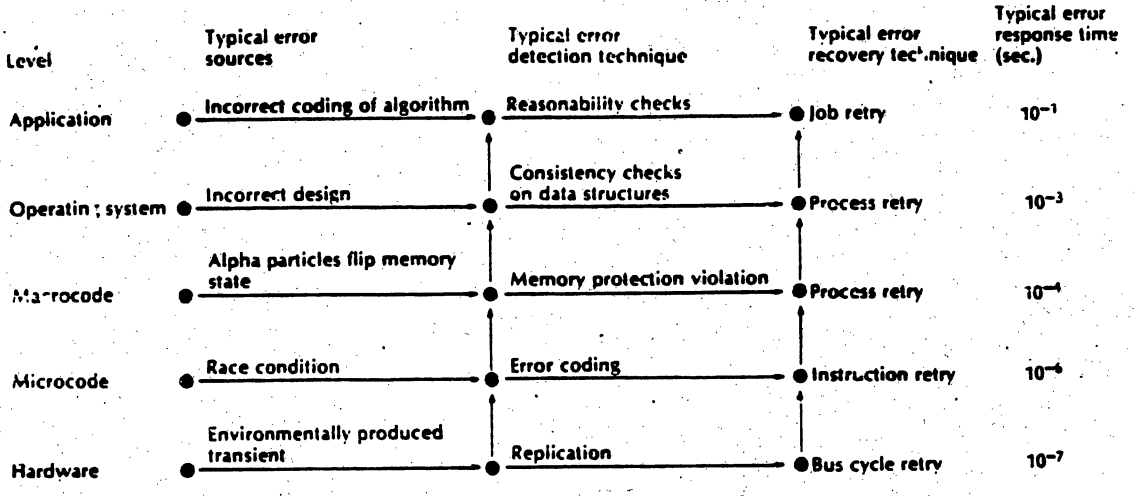


Figure 18–2. Levels in a hypothetical system.

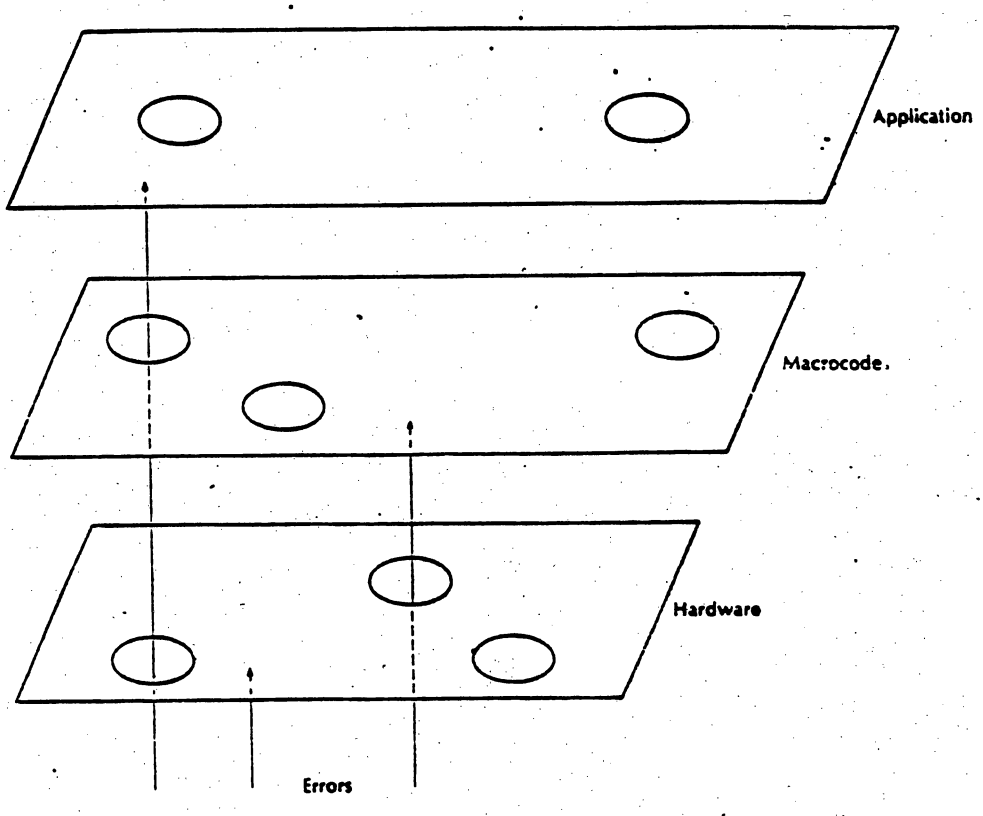


Figure 18–3. Interception of errors at multiple system levels due to imperfect coverage.

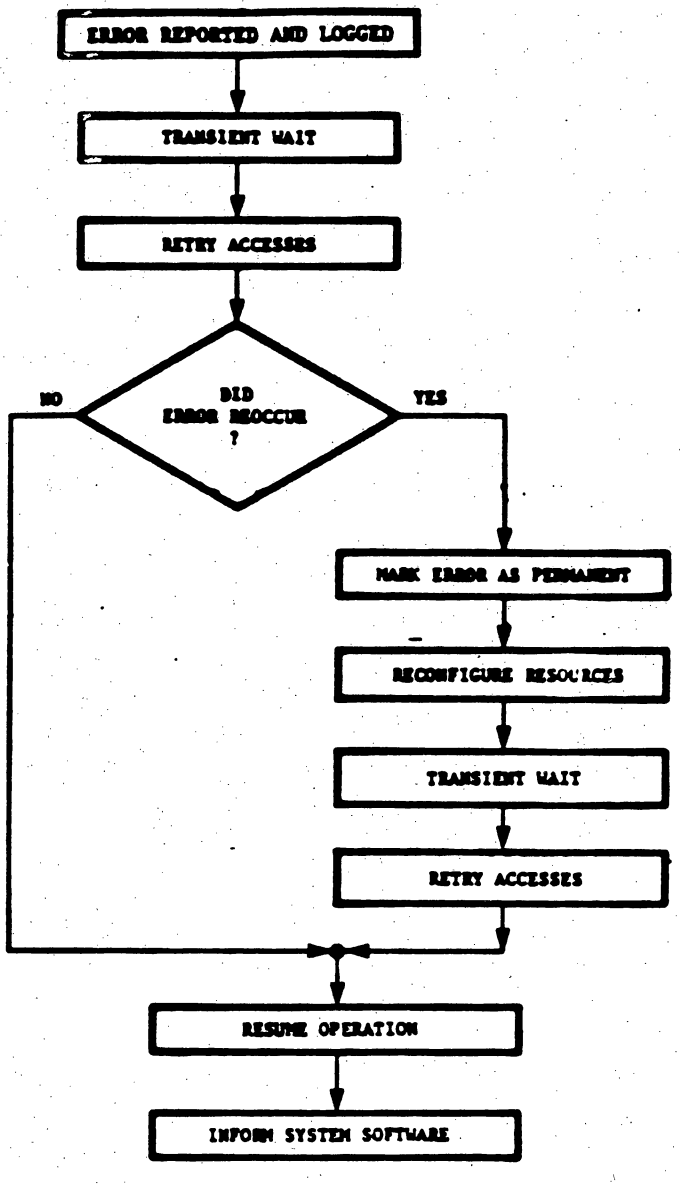


Figure 4-9. High-Level Recovery Sequence

P-0437

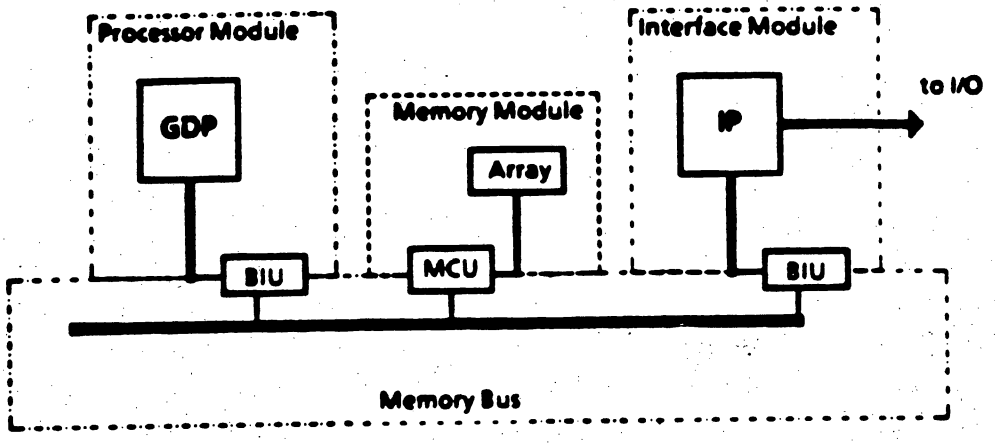


Figure 1-3. iAPX 432 Confinement Areas

F-0400

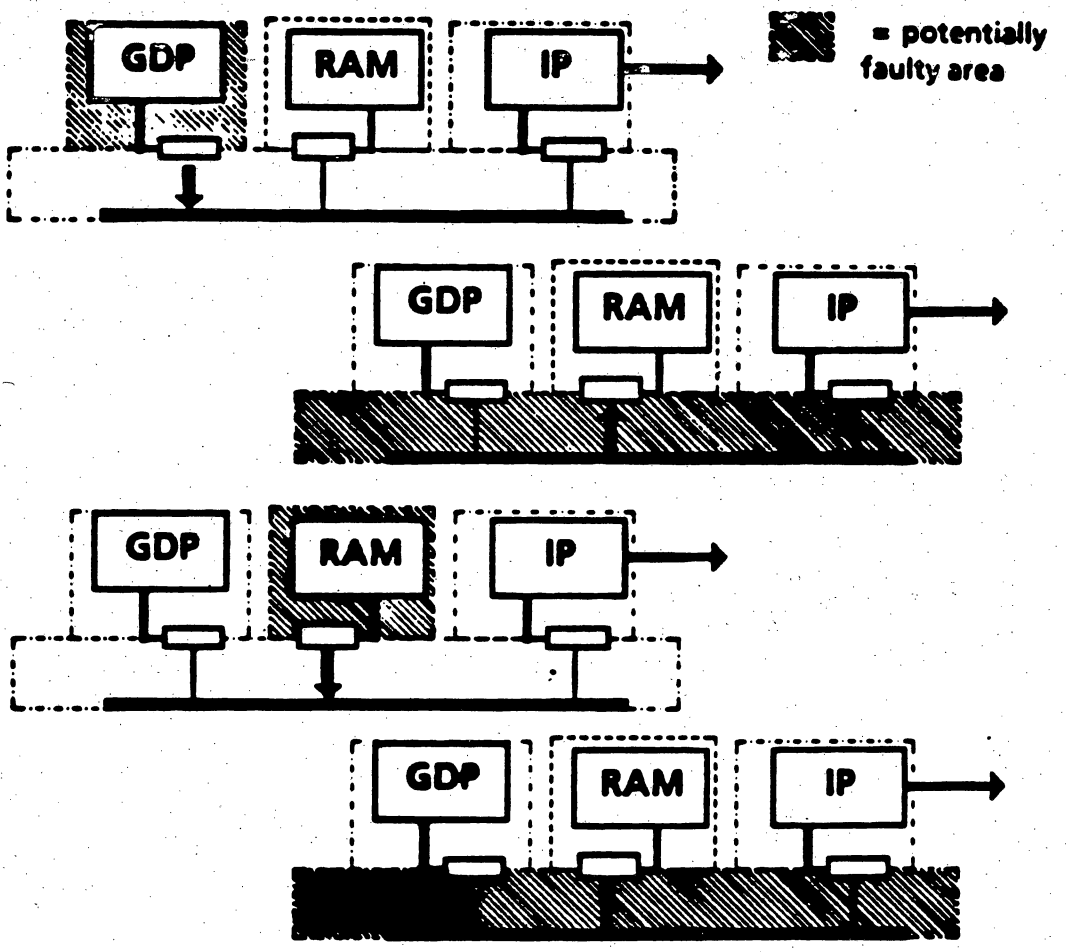


Figure 4-3. Confinement Area Operation

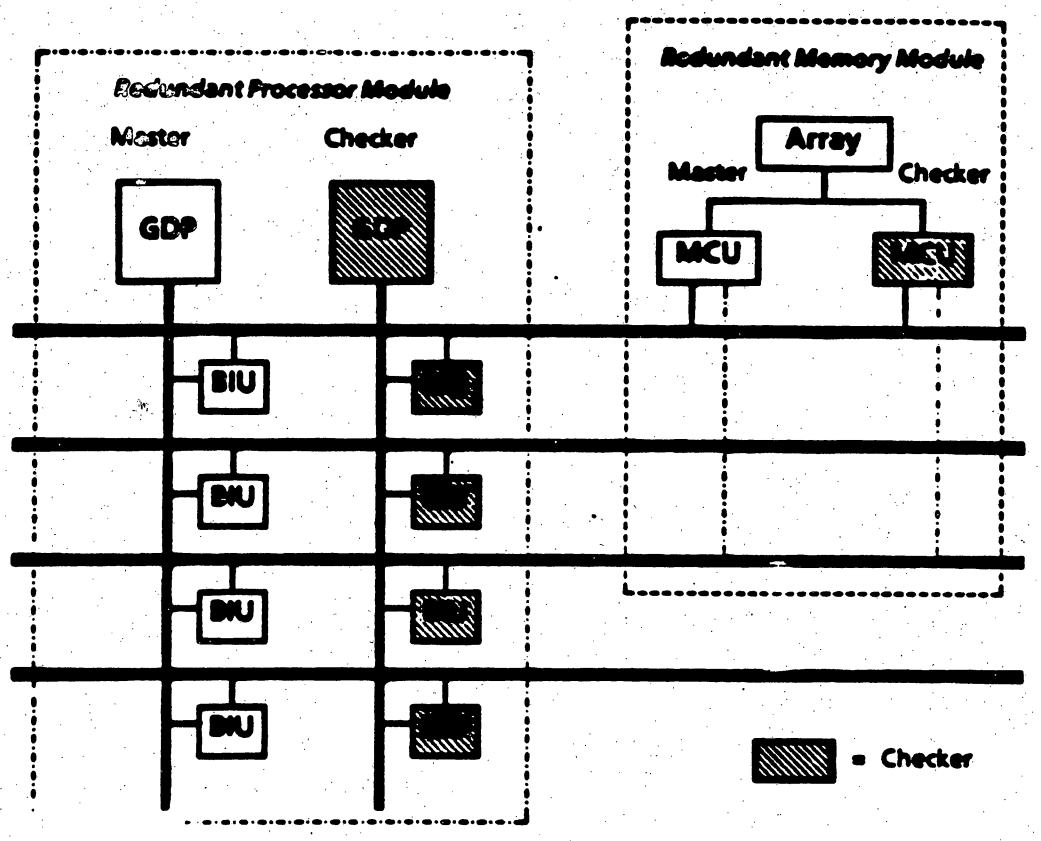


Figure 1-4. FRC Configuration Pairing

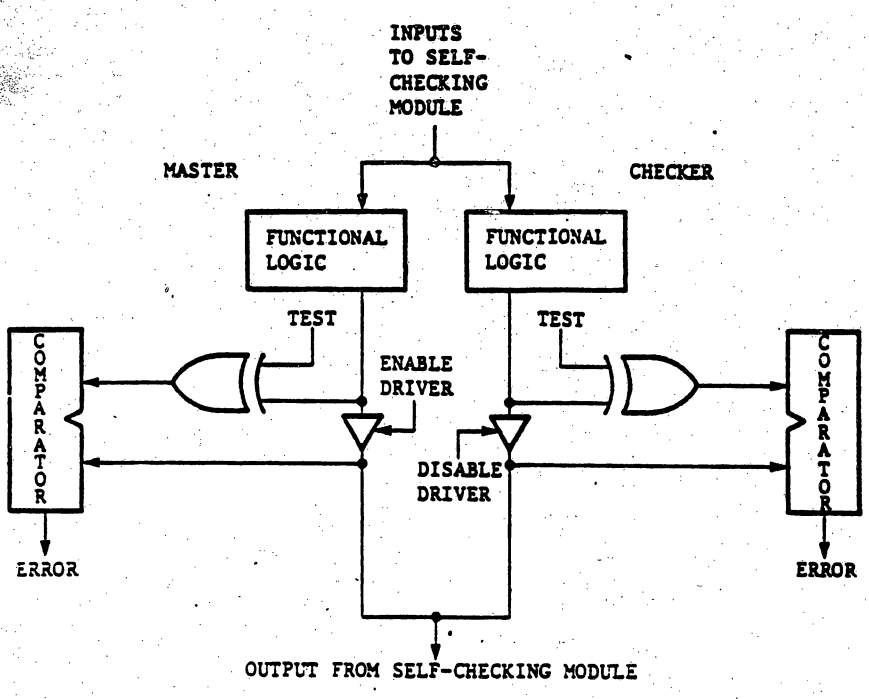


Figure 4-2. Functional Redundancy Checking

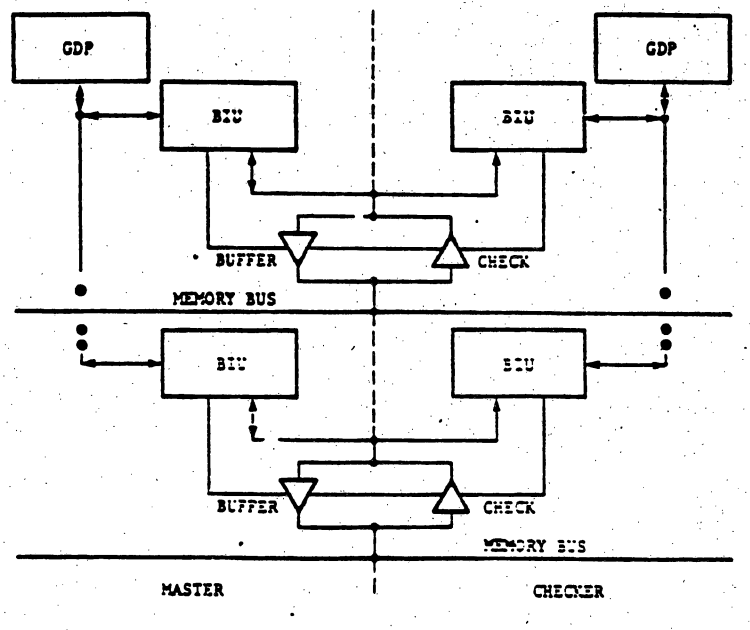
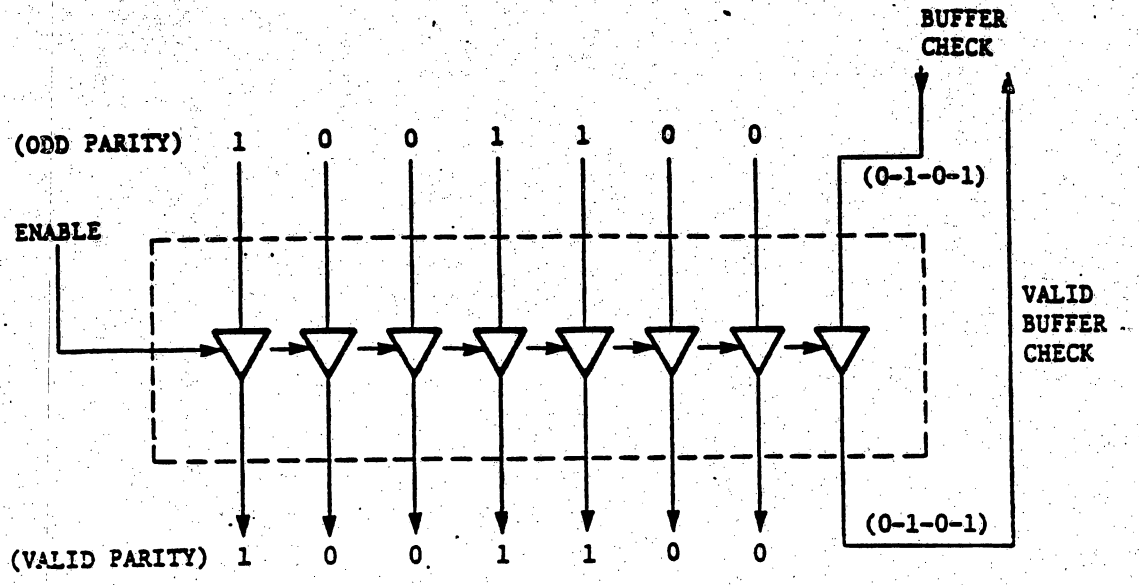


Figure 4-3. GDP Confinement Area



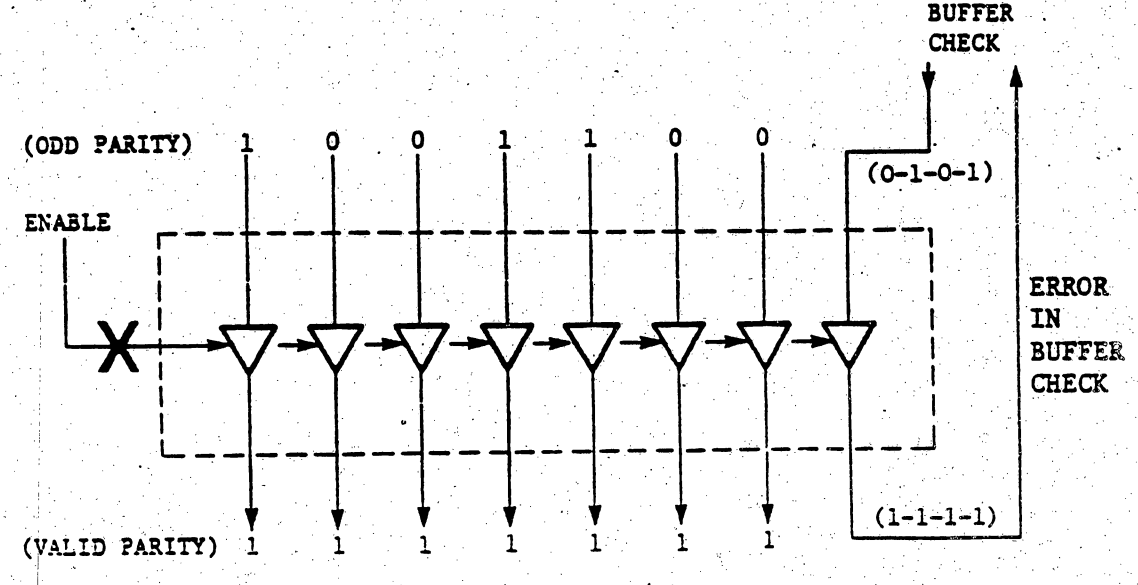


Figure 4-1. Buffer Check Detection Mechanism

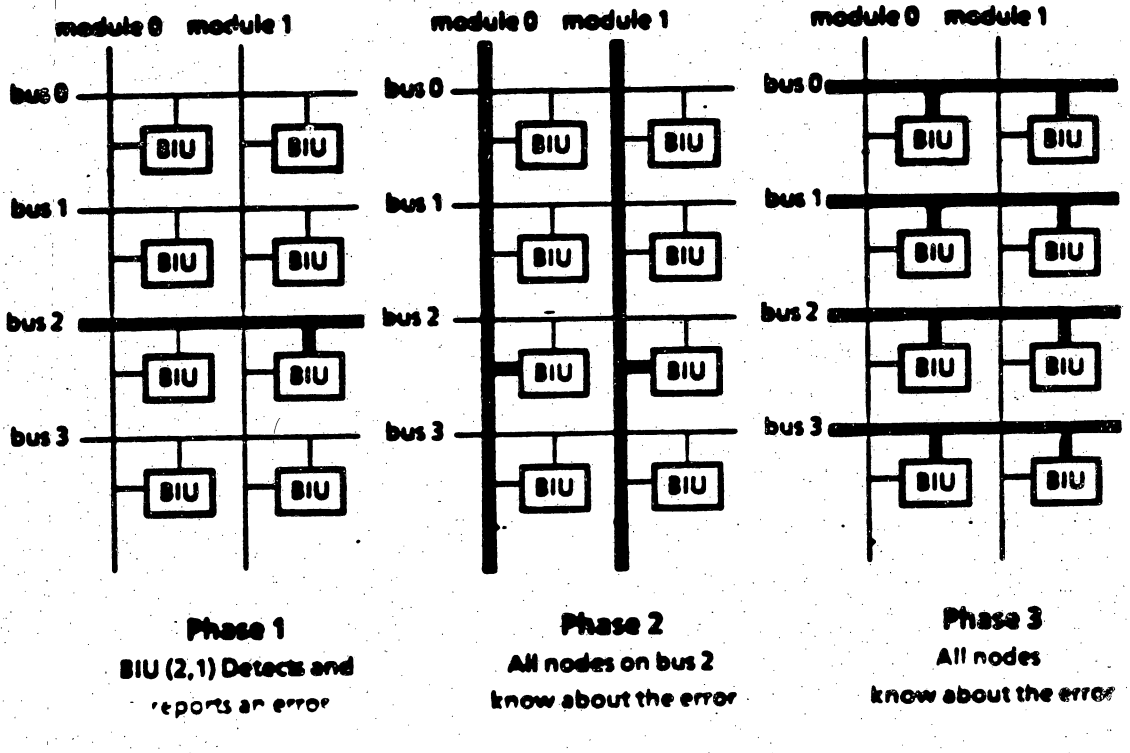


Figure 1-6. Three-Phase Error Reporting Mechanism

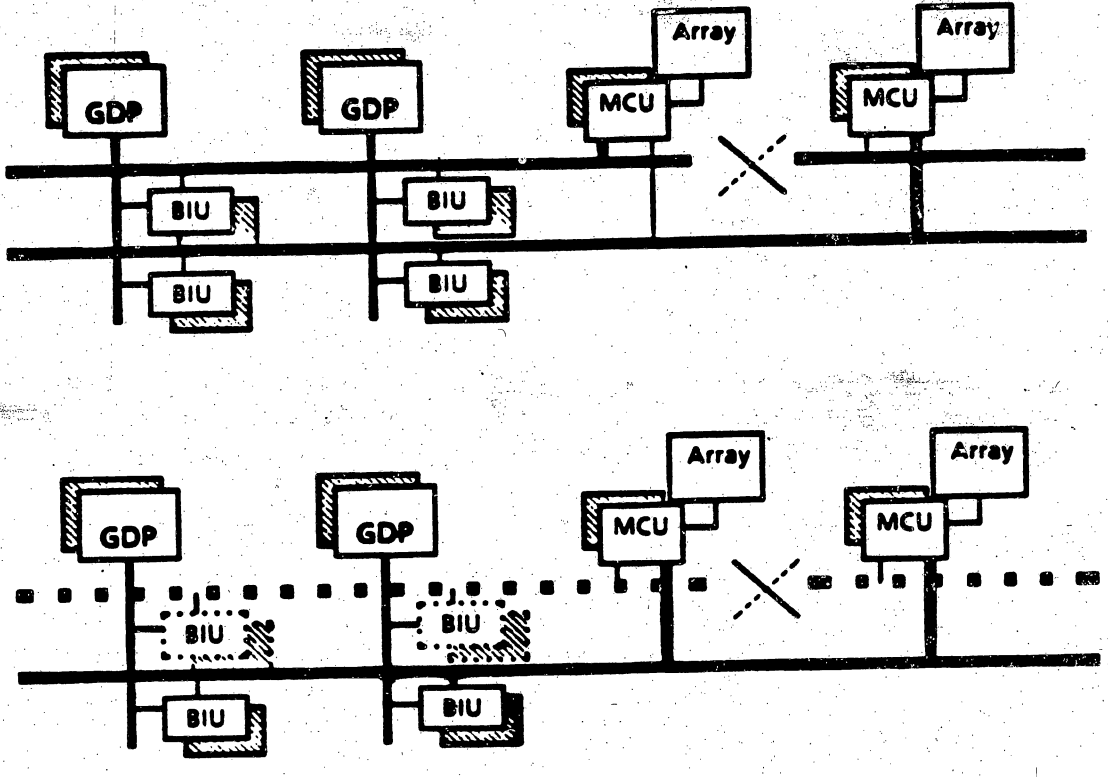


Figure 1-8. Bus Reconfiguration

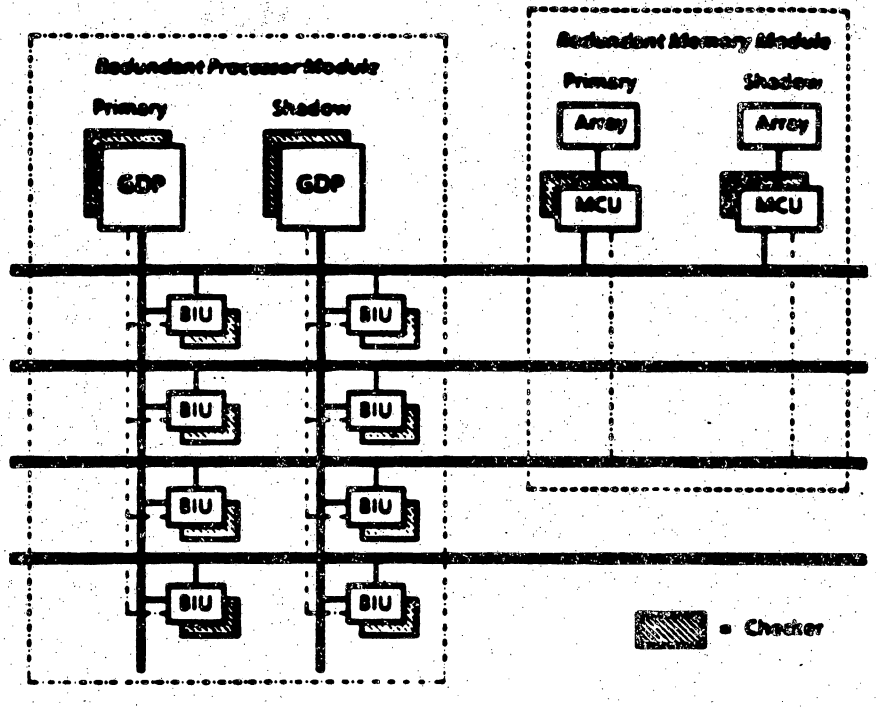


Figure 1-7. CMR Configuration Pairing

Table 1-2. Exercising Latent Faults

DORMANT AREA	EXERCISE
Memory Locations	MCU periodically reads every array location (scrubbing).
Detection Mechanisms	Software[a] periodically forces error conditions into the detection mechanisms.
Reporting Mechanisms	Software[a] periodically initiates and observes error reports.
Recovery Mechanisms	Software[a] periodically invokes recovery operations.
	[a]Special commands to support exercising dormant areas are provided in BIUs and MCUs.

Table 1-1. Recovery Mechanisms and Coverage

MECHANISM	COVERAGE
Retry	Transient Errors
ECC	Storage Array Address and Data
Spare Bit	DRAM Replacement
Memory Bus Pairs	Memory Bus Failure
Module Shadowing	Module Failure, GDP, IP, or Memory

Conclusion

- Designing from first principles to produce an architecture to tolerate failures achieves better reliability, availability, and cost effectiveness than an ad hoc, add-on approach.
- It is possible to build systems in which the activities of fault detection, diagnosis, and recovery are completely automated and transparent to the user.